User's Manual

AudioCodes Session Border Controller Series

Mediant 4000 SBC

Version 7.2

audiocodes

Notice

Date Published: January-23-2024

WEEE EU Directive

Pursuant to the WEEE EU Directive, electronic and electrical waste must not be disposed of with unsorted waste. Please contact your local recycling authority for disposal of this product.

Customer Support

Customer technical support and services are provided by AudioCodes or by an authorized AudioCodes Service Partner. For more information on how to buy technical support for AudioCodes products and for contact information, please visit our website at https://www.audiocodes.com/services-support/maintenance-and-support.

Documentation Feedback

AudioCodes continually strives to produce high quality documentation. If you have any comments (suggestions or errors) regarding this document, please fill out the Documentation Feedback form on our website at https://online.audiocodes.com/documentation-feedback.

Stay in the Loop with AudioCodes

Notes and Warnings

The device is an indoor unit and therefore, must be installed only **INDOORS**.

The scope of this document does not fully cover security aspects for deploying the device in your environment. Security measures should be done in accordance with your organization's security policies. For basic security guidelines, refer to the *Recommended Security Guidelines* document.

Configuration and usage of this AudioCodes device **must** be in accordance with your local security regulations, telephony regulations, or any other related regulations.

Throughout this manual, unless otherwise specified, the term *device* refers to your AudioCodes product.

Before configuring the device, ensure that it is installed correctly as instructed in the *Hardware Installation Manual*.

OPEN SOURCE SOFTWARE. Portions of the software may be open source software and may be governed by and distributed under open source licenses, such as the terms of the GNU General Public License (GPL), the terms of the Lesser General Public License (LGPL), BSD and LDAP. If any open source software is provided in object code, and its accompanying license requires that it be provided in source code as well, Buyer may receive such source code by contacting AudioCodes.

- Some of the features described in this document are licensed features and are available only if the installed License Key contains these features.

- This device includes software developed by the OpenSSL Project for use in the OpenSSL Toolkit (http://www.openssl.org/).
- This device includes cryptographic software written by Eric Young (eay@cryptsoft.com).

Related Documentation

Document Name
Release Notes
SBC-Gateway Series Release Notes for Latest Release Versions
SBC-Gateway Series Release Notes for Long Term Support Versions
Hardware / Installation Manuals

Document Name
MediaPack 504-508 (MP-5xx) Hardware Installation Manual
MediaPack 504-508 (MP-5xx) Voice Gateway Quick Guide
MediaPack 5xx Voice Gateway Basic Configuration
Mediant 4000B SBC Hardware Installation Manual
Mediant 4000B SBC Quick Guide
Complementary Guides
Gateway and SBC CLI Reference Guide
Mediant 4000 SBC SNMP Reference Guide
Recommended Security Guidelines
SIP Message Manipulation Syntax Reference Guide
Utility Guides
INI Viewer & Editor Utility User's Guide
CLI Wizard User's Guide

Document Revision Record

LTRT	Description
41727	Initial document release for Version 7.2.
41729	■ Updated patch version 7.20A.001. ■ Updated sections: Computer Requirements (supported browsers); CLI-Based Management (permitted user levels); Configuring TLS Certificate Contexts (TLS versions); Configuring Physical Ethernet Ports (show command); Configuring Underlying Ethernet Devices (max. VLANs); SIP Signaling Messages (procedure); First Incoming Packet Mechanism (NAT by Signaling); Robust Receipt of Media Streams by Media Latching (procedure); Configuring Firewall Settings (note); Configuring General Security Settings (Web path); Viewing IDS Alarms (show command); Viewing and Deleting DHCP Clients (show command); Configuring the Device's LDAP Cache; Centralized Third-Party Routing Server (SIP messages and credentials for authentication); Configuring Call Setup

LTRT	Description
	Rules (dial plan queries); Call Setup Rule Examples; Registration Refreshes; Using Dial Plan Tags for IP-to-IP Routing (example); Enabling Interworking of SIP and SIP-I Endpoints (SPIROU and SIP header X-AC-Action); Configuring WebRTC (Web path); Configuring BroadSoft's Shared Phone Line Call Appearance for Survivability (example); Automatic Provisioning (CLI Script file); Configuring RTCP XR (IP Group); Enabling Same Call Session ID over Multiple Devices (removed); Configuring Test Call Endpoints (typo); CLI (illustration); Accessing the Web Interface (note). ■ New sections: VoIPerfect; Using Dial Plan Tags for Call Setup Rules; Using Dial Plan Tags for Message Manipulation. ■ New parameters: WebLoginBlockAutoComplete; EnforcePasswordComplexity; IPGroup_SBCKeepOriginalCallID; IPGroup_SBCDialPlanName; IPGroup_CallSetupRulesSetId; CallSetupRules_QueryType; CallSetupRules_QueryTarget; IpProfile_SBCVoiceQualityEnhancement; IpProfile_SBCMaxOpusBW; IpProfile_SBCISUPVariant; AUPDCliScriptURL; PublicationIPGroupID. ■ Updated parameters: Web password (EnforcePasswordComplexity); TLSContexts_TLSVersion; InterfaceTable_InterfaceName; CallSetupRules_AttributesToQuery (Web name and description); IPProfile_SBCRTCPFeedback (values); IpProfile_MediaIPVersionPreference; ConditionTable_Name (max. chars); Test_Call_RouteBy (default); NATMode (values); SendAcSessionIDHeader (removed); QOEPort (removed); MaxGeneratedRegistersRate; CLIPrivPass; GeneratedRegistersInterval; RTPOnlyMode (removed); SBCUserRegistrationGraceTime; SBCKeepOriginalCallId
41731	■ Updated sections: Changing Index Position of Table Rows; Searching for Configuration Parameters; Configuring TLS Certificate Contexts (IPSec removed); Enabling the HTTP Proxy Application (license); Direct Media; Configuring SBC IP-to-IP Routing (IP Group load balancing); MAC Address Placeholder in Configuration File Name; VoIPerfect; Technical Specifications (AMR-WB removed). ■ New sections: Configuring IP Group Sets. ■ Updated parameters: SIPInterface_SBCDirectMedia; IPProfile_SBCDirectMediaTag; IpProfile_DisconnectOnBrokenConnection; IP2IPRouting_DestType; IPOutboundManipulation_PrivacyRestrictionMode; BrokenConnectionEventTimeout. ■ New parameters: IP2IPRouting_IPGroupSetName; EnableNonCallCdr; PGroupSet; IPGroupSetMember; NoRTPDetectionTimeout;

LTRT	Description
	DisconnectOnBrokenConnection; BrokenConnectionEventTimeout.
41733	■ Patch version 7.20A.100. ■ Updated sections: CLI (telnet removed); Areas of the GUI (SBC Wizard); Assigning Rows from Other Tables (search, add new, and view); Invalid Value Indications; Creating a Login Welcome Message; Configuring Management User Accounts (CLI); Enabling SSH with RSA Public Key for CLI (public key); Configuring TLS Certificate Contexts (DTLS); Assigning CSR-based Certificates to TLS Contexts; Generating Private Keys for TLS Contexts; Configuring Underlying Ethernet Devices (max.); Configuring IP Network Interfaces (max.); Configuring Media Realms (max.); SRTP using DTLS Protocol; Building and Viewing your Network Topology; SIP-based Media Recording (multiple SRSs); Enabling SIP-based Media Recording; Configuring SIP Recording Rules; Configuring Proxy Sets (keep-alive); WebRTC (RFCs); Configuring WebRTC; VoIPerfect; Pre-Configured IP Groups; Normal Mode (CRP); Emergency Mode (CRP); Auto Answer to Registrations (CRP); Network Topology Types and Rx/Tx Ethernet Port Group Settings; License Key; Viewing the License Key; Obtaining License Key for Feature Upgrade (removed); Installing the a New License Key; Installing License Key through Web Interface; Upgrading SBC Capacity Licenses by OVOC License Pool; Viewing Device Information; Viewing Call Routing Status (removed); Configuring RTCP XR (IP Group); Configuring RADIUS Accounting (typo for Accounting-Request); Automatic Provisioning (Startup CLI Script File). ■ New sections: Customizing the Web Interface; Replacing the Corporate Logo; Replacing the Corporate Logo with an Image; Replacing the Corporate Logo with Text; Customizing the Product Name; Customizing the Favicon; SRTP using DTLS Protocol; SBC Wizard; Viewing the Device's Product Key; Saving Configuration to a File; Loading a Configuration File; Viewing Proxy Set Status. ■ Updated parameters: TLSContexts_ServerCipherString; TLSContexts_ ClientCipherString; NATTranslation_SourceStartPort; NATTranslation_ SourceEndPort; NATTranslation_TargetStartPort; NATTranslation_ TargetEndPort; SNMPSysOid; SNMPTrapEnterpriseOid; EnableCoreDump (typo); HTTPSCipherString (removed); SSHAdminKey; SessionExpiresDisconnectTime; BrokenConnectionEventTimeout; RADIUSRetransmission (default); RadiusTO (default); SIPRecRouting_ RecordedIPGroupName; SIPRecRouting_SRSIPGroupName. ■ New parameters: WebUsers_SSHPublicKey; TLSContexts_DTLSVersion; TLSContexts_DHKeySize; SIPRecRouting_SRSRedundantIPGroupName; ProxySet_SuccessDetectionRetries; ProxySet_SuccessDetectionInterval;

LTRT	Description
	ProxySet_FailureDetectionRetransmissions; ProxySet_ MinActiveServersLB; WebUsers; WebFaviconFileUrl; AUPDStartupScriptURL.
41736	■ Updated sections: Configuring VoIP LAN Interface for OAMP (CLI); Configuring Management User Accounts (typo); Enabling SNMP; Configuring IP Network Interfaces; SIP-based Media Recording (multiple SRS); Configuring LDAP Servers (max. and cache); Configuring Call Setup Rules; Configuring SBC IP-to-IP Routing (note); Configuring SIP Response Codes for Alternative Routing Reasons; SBC Wizard (screens); Auxiliary Files (SBC Wizard); Viewing IP Connectivity (typo); Creating Core Dump and Debug Files upon Device Crash (reset); Configuring IP Group Sets (max) ■ New sections: Debugging Remote HTTP Services ■ Updated parameters: IpProfile_SBCUseSilenceSupp (removed): SIPRecRouting_SRSIPGroupName; SIPInterface_InterfaceName (max. char); ProxySet_ProxyName (max. char); MessageManipulations_ ManipulationName (max. char); MessagePolicy_Name (max. char); AllowedAudioCodersGroups_Name (max. char); AllowedVideoCodersGroups_Name (max. char); _ManipulationName (max. char); SBCAdmissionControl_AdmissionControlName (max. char); Classification_ClassificationName (max. char); IP2IPRouting_RouteName (max. char); SBCRoutingPolicy_Name (max. char); IPGroupSet_Name (max. char); IPInboundManipulation_ManipulationName (max. char); IPOutboundManipulation_ManipulationName (max. char); SBCAdmissionControl_Rate; EnableWebAccessFromAllInterfaces; ResetWebPassword; DisableSNMP; EnableCoreDump; SSHMaxLoginAttempts; IgnoreAlertAfterEarlyMedia; ECNLPMode; PremiumServiceClassMediaDiffServ; PremiumServiceClassControlDiffServ; IsFaxUsed; EnableAGC ■ New parameters: HTTPProxySyslogDebugLevel
41739	■ Updated with patch version 7.20A.150. ■ Updated sections: Areas of the GUI (Configuration Wizard button); Enabling Disabling SNMP; Viewing Certificate Information (screen); Assigning Externally Created Private Keys to TLS Contexts (pass-phrase); Generating Private Keys for TLS Contexts (pass-phrase); Importing Certificates into Trusted Certificate Store (bulk import); Configuring Underlying Ethernet Devices (MTU); Configuring Firewall Settings (note); SIP-based Media Recording (max.); Configuring Remote Web Services (QoS routing); Centralized Third-Party Routing Server (QoS); Configuring

LTRT	Description
	Proxy Sets; Alternative Routing Based on IP Connectivity; Configuring SBC IP-to-IP Routing; Configuring IP Group Sets (dial plan tags); Configuring Dial Plans; Software Upgrade; Installing License Key through Web Interface; Upgrading SBC Capacity Licenses by OVOC License Pool; SBC Configuration Wizard; Configuring RADIUS Accounting (typo); Configuring DTMF Tones for Test Calls; Configuring Basic Test Calls; Configuring SBC Test Call with External Proxy (removed). ■ New sections: Configuring QoS-Based Routing by Routing Server; Microsoft Skype for Business Presence of Third-Party Endpoints; Registrar Stickiness; Configuring Pre-Parsing Manipulation Rules; Configuring Private Wire Interworking; Configuring Rerouting of Calls to Fax Destinations; Using Dial Plan Tags for Routing Destinations; Disconnecting and Reconnecting HA; Viewing the License Key; Installing a License Key String; Viewing the Device's Product Key; Debugging Web Services. ■ Updated parameters: AccessList_Source_IP; AccessList_Source_Port; AccessList_Start_Port; AccessList_End_Port; HTTPRemoteServices_ HTTPType (option 5); IPGroup_SBCDialPlanName (note); ProxySet_ IsProxyHotSwap; ProxyIp_IpAddress; IpProfile_SBCRemoteReferBehavior (4); IpProfile_SBCPlayHeldTone; IP2IPRouting_Trigger (6); IP2IPRouting_ DestType (12/13); DialPlanRule_Tag; SBCCDRFormat_FieldType (818); Test_Call_RouteBy; Test_Call_Play (tone); KeepAliveTrapPort (default); SBCtestID (removed); ProxyIPListRefreshTime; RegistrationRetryTime (note); EnablePChargingVector (removed); EnableSBCApplication (default); SNMPReadOnlyCommunityString_x (max. char.); SNMPReadWriteCommunityString_x (max. char.); SNMPTrapCommunityString (max. char.). ■ New parameters: DeviceTable_MTU; SRD_SBCDialPlanName; SIPInterface_PreParsingManSetName; IPGroup_Tags; Account_ RegistrarStickiness; Account_RegistrarSearchMode; Account_ RegEventPackageSubscription; IpProfile_SBCFaxReroutingMode; IP2IPRouting_RoutingTagName; IP2IPRouting_InternalAction; IPGroupSet_Tags; CustomerSN; MaxRegistrationBackoffTime; MaxSDPSessionVersionId; UseRandomUser; UnregisterOnStartup; PresencePublishIPGroupId; EnableMSPresence; PreParsingManipulationSets; PreParsingManipulationRules; MWINotificationTimeout; RoutingServerQualityStatus; RoutingServerQualityStatusRate.
40201	■ Updated with patch version 7.20A.152. ■ Updated sections: Configuring the LDAP Search Filter Attribute (Web path); Enabling LDAP Searches for Numbers with Characters; Microsoft

LTRT	Description
	Skype for Business Presence of Third-Party Endpoints; Configuring the Device for Skype for Business Presence (example); Configuring Media Realm Extensions; Configuring Firewall Allowed Rules; Configuring SBC IP-to-IP Routing (back to the sender); Prerecorded Tones File; Installing on HA Devices (note); Loading a Configuration File (note) ■ Updated parameters: MediaRealmExtension_IPv4IF; MediaRealmExtension_IPv6IF; ProxySet_EnableProxyKeepAlive; SIPSDPSessionOwner ■ New parameters: IPGroup_SBCUserStickiness; IPProfile_ LocalRingbackTone; IPProfile_LocalHeldTone
42022	■ Updated with patch Version 7.20A.154.007 ■ Updated sections: Silence Suppression (removed); Fax / Modem Transparent Mode (silnce suppression removed); Configuring SIP Recording Rules (view sessions in CLI); Configuring RTP Base UDP Port (note removed re SIP Interface); Centralized Third-Party Routing Server (call preemption added); Locking and Unlocking the Device (typos); Viewing Active Alarms (max display) ■ New sections: Configuring Additional Management Interfaces; Configuring Specific UDP Ports using Tag-based Routing ■ Updated parameters: WebUsers_Password (note); InterfaceTable_ ApplicationTypes; CpMediaRealm_PortRangeStart (note removed); SIPInterface_UDPPort (note removed); ProxySet_SuccessDetectionRetries (max); ProxySet_SuccessDetectionInterval (max); Account_ RegistrarSearchMode (phys link); AudioCoders_Sce (global parameter removed); IpProfile_SCE (removed); IpProfile_SBCRemoteReferBehavior (new option 5); IPProfile_SBCRemoteHoldFormat (new option 6); IP2IPRouting_InternalAction; SBCCDRFormat_Title (max. char.); WebUsers (CLI name); EnableWebAccessFromAllInterfaces; FaxBypassPayloadType; ModemBypassPayloadType; EnableSilenceCompression (removed) ■ New parameters: SIPInterface_AdditionalUDPPorts; IPProfile_ SBCSupportMultipleDTMFMethods; AdditionalManagementInterfaces; DefaultTerminalWindowHeight; ActiveAlarmTableMaxSize; SBCRemoveSIPSFromNonSecuredTransport
42023	■ Updated with patch Version 7.20A.156.009 ■ Updated sections: Configuring Management User Accounts; Device Located behind NAT; Configuring a Static NAT IP Address for All Interfaces (removed); SIP-based Media Recording (URL of France reg.; note on SRS

LTRT	Description
	redundancy); Configuring SIP Recording Rules (note re timestamp); Configuring the OVOC Server (note re report mode); Configuring Call Setup Rules (ENUM); Call Setup Rule Examples (e.g., 5); Interworking SIP Early Media (figure); Prerecorded Tones File; Automatic Configuration Methods; DHCP-based Provisioning (note re resets) ■ New sections: Using Conditions for Starting a SIPRec Session; Using the Proprietary SIP X-AC-Action Header; Handling Registered AORs with Same Contact URIs; Configuring Dual Registration; Provisioning the Device using DHCP Option 160; Enabling SIP Call Flow Diagrams in OVOC ■ Updated parameters: WebUsers_SessionLimit; WebUsers_ SessionTimeout; SRD_BlockUnRegUsers (option 2 updated); SIPInterface_ AdditionalUDPPorts; SIPInterface_BlockUnRegUsers; IPGroup_ SIPConnect; CallSetupRules_QueryType (OPTION 3); CallSetupRules_ QueryTarget; CallSetupRules_AttributesToQuery; CallSetupRules_ Condition; IpProfile_SBCSDPPtimeAnswer; IpProfile_SBCPreferredPTime; IpProfile_SBCRemoteRepresentationMode (0 updated); DialPlanRule_ Tag; LoggingFilters_LogDestination (new option 3); LoggingFilters_ CaptureType (new option 6); WebSessionTimeout (range); StaticNatIP (removed); SBCDBRoutingSearchMode; SBCKeepContactUserinRegister ■ New parameters: WebUsers_CliSessionLimit; SIPRecRouting_ ConditionName; IPGroup_UserUDPPortAssignment; CallFlowReportMode; DhcpOption160Support; SIPRecTimeStamp ■ Miscellaneous: EMS/SEM replaced with One Voice Operations Center (OVOC) – text and screenshots
42025	■ Updated with Patch Version 7.20A.158 ■ Updated sections: Replacing the Corporate Logo with an Image (logo width removed); Replacing the Corporate Logo with Text; Customizing the Favicon (default); Creating a Login Welcome Message (no reset); Configuring Secured (HTTPS) Web; Disabling SNMP (reset); Configuring Underlying Ethernet Devices (Ethernet Output Device field); Configuring NAT Translation per IP Interface; Configuring Media (SRTP) Security (validation, no reset); Configuring Invalid RTP Packet Handling; LDAP-based Management and SIP Services (managed services); Configuring LDAP Server Groups (managed services); Configuring Registration Accounts; Configuring Rerouting of Calls to Fax Destinations; Configuring Call Preemption for SBC Emergency Calls (note removed); Configuring Call Survivability Mode (path); Configuring PSTN Fallback; Configuring Firewall Allowed Rules; Monitoring IP Entities and HA Switchover upon Ping Failure; Automatic Update from Remote Servers (AutoUpdatePredefinedTime); Viewing SBC CDR History; CDR Field

LTRT	Description
	Description; CDR Fields for SBC Signaling (removed); Configuring CDR Reporting; Storing CDRs on the Device; Enabling SIP Call Flow Diagrams in OVOC (note) ■ New sections: Restoring the Default Corporate Logo Image; Customizing the Browser Tab Label; Configuring SNMP for OVOC; Enabling Same Call Session ID over Multiple Devices; Customizing CDRs for Test Calls ■ Updated parameters: LdapServerGroups_ServerType (2); SIPInterface_ AdditionalUDPPorts; IPGroup_AuthenticationMode; ProxySet_ ProxyName (forward slash); IpProfile_TranscodingMode ; IP2IPRouting_ DestType (note); IP2IPRouting_DestIPGroupName (note); SBCCDRFormat_FieldType (442, 635); TelnetServerEnable; DisableSNMP; EnableLanWatchDog (removed); CDRLocalMaxFileSize (name change, max.); CDRLocalMaxNumOfFiles (name change); CDRLocalInterval (name change); SyslogOptimization (def); HAPingEnabled; HAPingDestination (removed); HAPingSourceIfName (removed); HAPingTimeout (removed); HAPingRetries (removed); EnableMediaSecurity (no reset) ■ New parameters: Account_RegByServedIPG; Account_ UDPPortAssignment; IpProfile_SBCAdaptRFC2833BWToVoiceCoderBW; TimeZoneFormat; CallDurationUnits; SendAcSessionIDHeader; HaNetworkMonitorThreshold; HaNetworkMonitor; SRTPTunnelingValidateRTPRxAuthentication; SRTPTunnelingValidateRTCPRxAuthentication; RTPFWInvalidPacketHandling; RtpFWNonConfiguredPTHandling ■ Miscellaneous: New AudioCodes logo; '.content' removed fom manipulation syntax
42026	■ Updated to Patch Version 7.20A.200.019 ■ Updated Sections: Default OAMP IP Address; Configuring Management User Accounts (max.); Assigning CSR-based Certificates to TLS Contexts (SAN); Creating Self-Signed Certificates for TLS Contexts (SAN); DNS; Configuring Remote Web Services (Capture removed); HTTP-based Proxy Services; Debugging Remote HTTP Services; Configuring an HTTP-based OVOC Service; Configuring User Information; Configuring SBC User Info Table from a Loadable File; Configuring SIP Message Manipulation (max.); Interworking Media Security Protocols; Enabling the SBC Application (removed); Configuring Admission Control; Enabling the CRP Application (removed); Configuration while HA is Operational (resets); Auxiliary Files (User Info): User Information File; Viewing the License Key; Upgrading SBC Capacity Licenses by OVOC License Pool; Restoring Factory Defaults through CLI (keep-network); Viewing Active Alarms (refresh and order); Viewing History Alarms (refresh and order); Viewing Proxy Set Status

LTRT	Description
	(name); Configuring RTCP XR (note); Configuring Reporting of Management User Activities (select all); Configuring Test Call Endpoints (max.) ■ New Sections: Configuring Default DNS Servers; Configuring a DNS Server for HTTP Services; Configuring HTTP Proxy Servers; Configuring HTTP Locations; Configuring TCP/UDP Proxy Servers; Configuring Upstream Groups; Configuring Upstream Hosts; Configuring HTTP Directive Sets; Configuring HTTP Directives; Troubleshooting NGINX; Configuring SBC MoH from External Media Source; Saving and Loading an ini Configuration to a File; Viewing IDS Active Blacklist; Saving and Loading a Configuration Package File; Quick-and-Easy Initial HA Configuration; Backing Up and Restoring HA Configuration ■ Updated Parameters: WebUsers_SessionLimit (max/def); HTTPRemoteServices_HTTPType (Capture removed); CallSetupRules_ AttributesToQuery; CallSetupRules_Condition; CallSetupRules_ ActionSubject; CallSetupRules_ActionValue; MessageManipulations_ MessageType; MessageManipulations_Condition; MessageManipulations_ActionValue; ConditionTable_Condition; PreParsingManipulationRules_MessageType; PreParsingManipulationRules_ReplaceWith; AudioCoders_Sce; IpProfile_ SBCMediaSecurityMethod (2 removed); IPProfile_ SBCRemoteHoldFormat; IPProfile_ReliableHoldToneSource; IpProfile_ SBCPlayHeldTone; IP2IPRouting_InternalAction; MaliciousSignatureDB_ Pattern; SSHServerEnable (def); NoRTPDetectionTimeout; SBCAdmissionControl (removed); DialPlanRule; HTTPProxySyslogDebugLevel (options); HARevertiveEnabled; HAPriority; HARemotePriority; MediaChannels ■ New Parameters: SRD_AdmissionProfile; MessageManipulations_ ActionSubject SIPInterface_AdmissionProfile; IPGroup_AdmissionProfile; SBCAdmissionRule_MaxBurstPerUser; Rate Per User; SBCUserInfoFileUrl; UserInfoFileURL; ConfPackageURL; DefaultPrimaryDnsServerIp; DefaultSecondaryDnsServerIp; MaxStreamingCalls; SBCAdmissionProfile; SBCAdmissionRule; SBCUserInfoTable; ExternalMediaSource; HTTPPrimaryDNS; HTTPSecondaryDNS; HTTPServer; HTTPLocation; TcpUdpServer; UpstreamGroup; UpstreamHost; HTTPDirectiveSets; HTTPDirectives; OVOCService; HALocalMAC; HARemoteMAC; MaxStreamingCalls
42028	■ Updated to Patch Version 7.20A.200.112 ■ Updated Sections: Accessing the Web Interface (remember me); Configuring Additional Management Interfaces (CLI removed);

LTRT	Description
	Configuring TLS Server Certificate Expiry Check (acCertificateExpiryAlarm); Fax / Modem NSE Mode (removed); Configuring SIP Recording Rules; Building and Viewing SIP Entities in Topology View (access via logo); Remotely Resetting Device using SIP NOTIFY (SBC also); VoIPerfect (Managed G.729); Locking and Unlocking the Device (cancel); Software Upgrade (fallback); License Key (floating); Viewing the License Key (floating / Device Type / License Server Status / Remote License Server Status - deleted / Remote License Server IP - deleted); Local License Key; SBC Capacity Licenses from Fixed License Pool; Saving and Loading an ini File (License Key of Redundant); Saving and Loading a Configuration Package File (SFTP); Triggers for Automatic Update (NOTIFY for SBC also); Viewing Active Alarms (HA / format); Viewing History Alarms (HA / format); CDR Field Description (Alerting Time [443]); Patterns for Denoting Phone Numbers and SIP URIs ■ New Sections: Remotely Resetting Device using SIP NOTIFY; Remotely Disconnecting Calls using SIP NOTIFY; SBC Capacity Licenses from Floating License; Applying Downloaded ini File after Graceful Timeout; Debug Recording on VoIP Interfaces ■ Updated Parameters: SNMPUsers_PrivProtocol (4 and 5 removed); InterfaceTable_InterfaceName (default); "Prefix" replaced with "Pattern" for Web parameters and CLI commands and descriptions updated; CpMediaRealm_MediaRealmName (range); CpMediaRealm_ PortRangeStart (note); MediaRealmExtension_PortRangeStart (note); SIPInterface_UDPPort (note); SIPInterface_TCPPort (note); SIPInterface_ TLSPort (note); SIPInterface_AdditionalUDPPorts (note); IPGroup_ ContactUser (val); IPGroup_Username (max. val); Account_ContactUser (max. val); SBCUserInfoTable_LocalUser (max value); SBCUserInfoTable_ Username (max. val); IpProfile_SBCMediaSecurityMethod (Both value added); IpProfile_SBCRemoteUpdateSupport (option 3 added); SBCAdmissionRule_RequestType; IP2IPRouting_DestType (Internal rules); SBCCDRFormat_FieldType (443); Test_Call_UserName (max. values); Test_Call_Play (note removed); UserName (max. val); UseRandomUser (description); MgmtUseLocalUsersDatabase (description); EnableSIPRemoteReset; SBCRandomizeExpires (description); TLSExpiryCheckStart; MediaChannels ■ New Parameters: IpProfile_SBCRemoveUnKnownCrypto; DisableDualImageFeature; AdminStateLockControl; EnableFloatingLicense; AllocationProfile; AllocationRegisteredUsers; AllocationMediaSessions; AllocationSignalingSessions; LimitRegisteredUsers; LimitMediaSessions; LimitSignalingSessions; LimitTranscodingSessions; AupdGracefulShutdown; MPLSMode;

LTRT	Description
	RetryAfterMode; SIPDigestAuthorizationURIMode; EnableSIPRemoteReset
42029	■ Updated to Patch Version 7.20A.204.015 ■ Updated Sections: TLS for SIP Clients (no reset); Configuring Physical Ethernet Ports; First Incoming Packet Mechanism; Configuring TLS for SIP; Enabling IDS (no reset); SIP-based Media Recording (RFC s updated); Enabling SIP-based Media Recording (removed); Configuring Proxy Sets; Configuring Call Setup Rules (max.); Supported Audio Coders (G.722); Alternative Routing Based on SIP Responses; Classification and Routing of Registered Users; Configuring Call Preemption for SBC Emergency Calls; Configuring Firewall Allowed Rules; Locking and Unlocking the Device (TLS sockets); Installing on HA Devices; SBC Capacity Licenses from Floating License; Viewing Average Call Duration; Event Representation in Syslog Messages; Creating Core Dump and Debug Files upon Device Crash ■ New Sections: Configuring CSRF Protection; Configuring Firewall Rules to Allow Incoming OVOC Traffic; Configuring Format of SIPRec Metadata; Configuring an HTTP GET Web Service; Configuring Classification Based on Tags; Accessing Files on Redundant from Active Device through SSH; Viewing Floating License Reports; Configuring CDR Reporting to REST Server ■ Updated Parameters: EtherGroupTable_Mode; HTTPRemoteServices_ HTTPType (8); HTTPRemoteServices_Policy; HTTPRemoteServices_ PersistentConnection; HTTPRemoteServices_NumOfSockets (removed); HTTPRemoteServices_LoginNeeded (removed); IPGroup_Tags; ProxySet_ EnableProxyKeepAlive; ProxySet_ProxyRedundancyMode; ProxySet_ IsProxyHotSwap; ProxySet_ProxyLoadBalancingMethod; ProxySet_ DNSResolveMethod; CallSetupRules_RulesSetID; CallSetupRules_ QueryType (4); CallSetupRules_QueryTarget; IPGroupSet_Tags; IPOutboundManipulation_SrcTags; TelnetServerIdleDisconnect; SNMPEngineIDString; NATMode (4); SIPSRequireClientCertificate (no reset); EnableIDS (no reset); EnableSIPRec (removed); IpProfile_ SBCRFC2833Behavior; IpProfile_SBCAlternativeDTMFMethod ■ New Parameters: HTTPRemoteServices_BetweenGroupsPolicy; AdditionalUDPPortsMode; SIPInterface_CallSetupRulesSetId; ProxyKeepAliveUsingIPG; ProxyIp_Priority; ProxyIp_Weight; Classification_IPGroupSelection; Classification_IpGroupTagName; AdminState; AdminStateRestrictConnections; CSRFProtection; RestCdrReportLevel; RestCdrHttpServer; SSHRedundantProxyPort; SBCURIComparisonExcludedParams; SbcRoutingTimeout; SIPRecMetadataFormat; HTTPRemoteHosts_GroupID;

LTRT	Description
	HTTPRemoteHosts_PriorityInGroup
42030	■ Updated to Patch Version 7.20A.204.108 ■ Updated Sections: Enabling SSH with RSA Public Key for CLI (Admin key removed); Interfaces Table Configuration Guidelines (31); Device Located behind NAT; Configuring Firewall Rules to Allow Incoming OVOC Traffic (note); Enabling the User Info Table; Configuring SBC User Information (note); Call Setup Rule Examples (ENUM); Using the Proprietary SIP X-AC-Action Header (REFER); Storing CDRs on the Device (HA note) ■ New Sections: Configuring a Static NAT IP Address for All Interfaces (returned) ■ Updated Parameters: WebUsers_Password (\); WebUsers_SSHPublicKey; InterfaceTable_PrefixLength (31); InterfaceTable_Gateway; IPGroup_ AuthenticationMode; IPGroup_Username; IPGroup_Password; ProxySet_ ProxyRedundancyMode (note); CallSetupRules_QueryType (ENUM address); CallSetupRules_QueryTarget (ENUM); IpProfile_ SBCRemoteReferBehavior; IP2IPRouting_DestType (note); SSHAdminKey (removed); SSHRequirePublicKey; DisableICMPUnreachable ■ New Parameters: StaticNatIP; ForceGenerateToTag; LdapConfiguration_ NoOpTimeout
42031	■ Updated sections: Configuring Voice Mail; Call Processing of SIP Dialog Requests; RADIUS-based Management User Authentication (CLI); Enabling LDAP-based Web/CLI User Login Authentication and Authorization; Media Cluster (time); Collecting Debug Recording Messages; Viewing SBC Registered Users; Interworking SIP Diversion and History-Info Headers ■ Updated parameters: IpProfile_SBCISUPBodyHandling (new value); LogoWidth (removed); LocalStorageMedia (removed- only SD card storage) ■ New parameters: GWInboundManipulationSet; GWOutboundManipulationSet
42033	Updated for software update Version 7.20A.250. ■ Updated sections: Alternative Routing Based on SIP Responses (806 updated); Fax and Modem Capabilities (removed); CDR Field Description (SIP Local/Remote Tag); CLI-Based Management (user levels); Configuring Charge Codes (CLI command); Accessing Web (Log In button name); Configuring Management User Accounts (user level descriptions); Configuring Media (SRTP) Security (AES-256); Configuring SIP Recording

LTRT	Description
	Rules (note for SRTP-SRTP recording); Configuring the Internal DNS Table (note); Configuring the Internal SRV Table (max); Configuring Default DNS Servers; Configuring the Trunk Group Settings Table (max); Event Detection and Notification using X-Detect Header (note re CPT); HTTP Proxy Parameters (CLI); Monitoring IP Entities and HA Switchover upon Ping Failure (hostname); Prerecorded Tones File ("acUserDefine" and description); Restoring Factory Defaults through Web Interface (check box name); Storing CDRs on the Device (note re HA); Viewing Device Information (device uptime format);WebRTC (sessions); Configuring HTTP Directive Sets (values changed of limit_conn and limit_rate); File Location for Automatic Update (username-password); ■ New sections: Configuring a Public IP Address for NGINX NAT Traversal; Configuring SIP Response Codes to Exclude from IDS; ■ Updated parameters: CliScriptURL (typo); CLIStartupScriptUrl (typo); AccessList_Protocol ("sip" removed); IDSRule_Reason (CAC, exclusion of SIP cause codes); IPGroup_SIPConnect (options and descriptions); IPGroup_MethodList ("setup-invite"); IpProfile_SBCRemote3xxBehavior (new options 3 and 4); IPProfile_LocalRingbackTone (PRT userdefine); IPProfile_LocalHeldTone (PRT userdefine); LoggingFilters_CaptureType (option 7 added, note added for option 4); WebUsers_PwAgeInterval (description and note); ProxySet_ClassificationInput (note); ProxySet_ClassificationInput (note and example); ProxySet_DNSResolveMethod (max. hostnames); ProxyIp_IpAddress (note for IP addresses); HTTPRemoteServices_KeepAliveTimeOut (description and note removed); RTCPInterval (Web name changed); SIPInterface_PreClassificationManipulationSet (note if classification fails); SBCCDRFormat_FieldType (445 and 446 added); GWInboundManipulationSet (CLI); GWOutboundManipulationSet (CLI); HaNetworkMonitor_DestAddress (hostname); NTPServerUTCOffset (range); UdpPortSpacing (default); T38FaxMaxBufferSize (default); VQMonEnable (reset M1K); NumOfSubscribes (note removed); SBCServerAuthMode (note re IPGroup_TypeSBCServerAuthType); SRTPofferedSuites (AES-256); SSHRedundantProxyPort (CLI); TelnetServerEnable (note); AUPDDigestUsername (removed); AUPDDigestPassword (removed); Account_RegistrarStickiness; IP2IPRouting_RequestType (note); ActivityListToLog (ard removed) ■ New parameters: CallSetupRules_RulesSetName; IPGroup_TypeSBCServerAuthType; IPGroup_OAuthHTTPService; Account_AccountName; SBCAlternativeRoutingReasons_AltReasonName; PPreferredIdListMode; HttpProxyGlobalAddress; DialPlanCSVFileUrl; Account_ReRegisterOnInviteFailure; AccountInviteFailureTriggerCodes

LTRT	Description
42034	Updated to Patch Version 7.20A.252.011. ■ Updated sections: Accessing Files on Redundant Device from Active through SSH (note added); Areas of the GUI (password on toolbar and hostname); Assigning CSR-based Certificates to TLS Contexts; Assigning Externally Created Private Keys to TLS Contexts (Status field); Assigning IDS Policies (fields renamed); CDR Field Description (Media List and Call Success added); Centralized Third-Party Routing Server (Call Forking added); Configuring Call Admission Control (note); Configuring Basic Test Calls (note); Configuring Call Setup Rules (HTTP Post Notification); Configuring CDR Filters and Report Level (CDR End parameters); Configuring Classification Rules (capacity); Configuring Coder Groups (note re G.729ab); Configuring Dial Plans (capacity and range); SBC Configuration Wizard (not for HA); Configuring Ethernet Port Groups (device reset); Configuring Physical Ethernet Ports (device reset); Configuring Gateway User Information Table through Web Interface (table name); Configuring ICMP Messages (parameter names); Configuring IDS Policies (editable defaults/WebSocket failures); Configuring IP Groups (capacity); Configuring Management User Accounts; Configuring Password Display in ini File; Configuring Proxy Sets (edit and capacity); Configuring SBC Routing Policy Rules (capacity); Configuring SIP Interfaces (capacity); Configuring SIP Response Codes for Alternative Routing Reasons; Configuring SNMP Trap Destinations with IP Addresses; Configuring SRDs (capacity); Configuring Static IP Routes (reset removed); Configuring Syslog Debug Level (typo); Configuring Test Call Endpoints; Configuring the Device's LDAP Cache (typo); Configuring the OVOC Server for QoE (table renamed); Configuring the Syslog Server Address (path); Configuring TLS Certificate Contexts;Connectivity and Synchronization between Devices; Creating Core Dump and Debug Files upon Device Crash (note); Creating Self-Signed Certificates for TLS Contexts (CN); CRP Configuration (note); ; Customizing CDRs for SBC Calls and Test Calls (JSON); Dial Plan (backward compatibility sections removed); Enabling SIP Call Flow Diagrams in OVOC; Enabling Syslog (path); Initial Registration Request Processing; Installing a License Key String; Network Topology Types and Rx/Tx Ethernet Port Group Settings; No-Op Packets (DSP note removed); Saving and Loading CLI Script Files (optional reset); Saving and Loading a Configuration Package File; Saving and Loading an ini Configuration File; SBC Capacity Licenses from Floating License (note on ignore licenses); Viewing IDS Alarms (WebSocket failures); Viewing Logged-In User Information; Changing OAMP Address through Web Interface ■ New sections: Changing Login Password by All User Levels; Configuring a Hostname for the Device; Configuring DiffServ for HA Maintenance

LTRT	Description
	Traffic; Configuring Password Display in CLI; Configuring User-Defined Performance Monitoring MIBs; Configuring Syslog Message Severity Level; Customizing Access Levels per Web Page; Customizing SNMP Alarm Severity; Exporting Dial Plans; Idle CLI Session Timeout for RS-232 Connections; Importing Dial Plans; Miscellaneous CDR Configuration; Syslog Message Description for CPU Overload; Viewing CDR History of SBC and Test Calls ■ New parameters: MatrixCsvFileUrl; IpProfile_CreatedByRoutingServer; HTTPRemoteServices_LoginNeeded; HTTPRemoteServices_ VerifyCertificateSubjectName; Test_Call_OfferedCodersGroupName; Test_Call_AllowedAudioCodersGroupName; Test_Call_ AllowedCodersMode; Test_Call_MediaSecurityMode; Test_Call_ PlayDTMFMethod; Test_Call_MediaSecurityMode; QOESettings_ VerifyCertificateSubjectName; Hostname; HAMaintenanceIFDiffServValue; HAOperationalStateDelayInSec; ShortCallSeconds; SyslogLogLevel; CallEndCDRSIPReasonsFilter; CallEndCDRZeroDurationFilter; CLIEnableModePassword; CliObscuredPassword ■ Updated parameters: CallSetupRules_QueryType (HTTP POST Query / HTTP POST Notification); CallSetupRules_QueryTarget (Web name changed); CallSetupRules_AttributesToQuery (Web name changed); CallSetupRules_ActionType (None value added); InterfaceTable_ PrefixLength (values); IpProfile_SBCRemoteReferBehavior (Keep URI (user@host)); IpProfile_AMDMaxPostSilenceGreetingTime (default); LoggingFilters_Value (Any removed); LoggingFilters_CaptureType (note added to 2 and 4); WebUsers_Password (note); IP2IPRouting_DestTyp (Gateway value update); SNMPTrapCommunityStringPassword (name change); SNMPReadWriteCommunityStringsPassword (name change); SNMPReadOnlyCommunityStringsPassword (name change);Test_Call_ DestTransportType (SCTP added); Test_Call_PlayDTMFMethod (In Band value); Test_Call_Play (PRT and NetAnn); QOESettings (parameters renamed); TLSContexts_ServerCipherString (default); TLSContexts_ ClientCipherString (default); ntpAuthMd5KeyPassword (renamed); PM_ EnableThresholdAlarms
42036	Updated to Patch Version 7.20A.254.202. ■ Updated sections: Configuring Allowed Video Coder Groups (typo); Filtering IP Network Traces (IPv6); Starting the SBC Configuration Wizard (screenshot and typos); CDR Field Description (codes for termination reasons; RELEASE_BECAUSE_ACCOUNT_NOT_REGISTERED removed); Configuring Call Setup Rules (False/True value); Customizing the Product Name (no reset); Configuring TLS for SIP (figure updated); Customizing

LTRT	Description
	CDRs for SBC Calls and Test Calls (max. RADIUS attributes); Configuring SIP Response Codes for Alternative Routing Reasons; Configuring Message Session Relay Protocol ■ New sections: Customizing CDR Indication for Call Success or Failure based on Responses; Configuring Video Recording Synchronization; Enabling DNS Rebinding Protection ■ Updated parameters:IpProfile_SbcPrackMode (new options 0 and 4); SIP Group Name; TLSContexts_TLSVersion (option [5] TLSv1.0 and TLSv1.2 removed); IsUserPhone (removed); NoOpInterval (range and default); ShortCallSeconds (Web parameter); InterfaceTable_PrefixLength (range for IPv6); IpProfile_SBCMediaSecurityBehaviour (values renamed); IPGroup_DTLSContext (Web name change); ■ New parameters: IPGroup_TopologyHidingHeaderList; CallSuccessSIPReasons; CallFailureSIPReasons; CallSuccessInternalReasons; CallFailureInternalReasons; NoUserResponseBeforeConnectSuccess; NoUserResponseAfterConnectSuccess; CallTransferredBeforeConnectSuccess; CallTransferredAfterConnectSuccess; VideoRecordingSyncTimeout; SIPInterface_SCTPPort; SBCAltRoutingReasonsSet; SBCAltRoutingReasonsList; IPGroup_SBCAltRouteReasonsSetName; ; IpProfile_SBCGenerateNoOp; DNSrebindingProtectionEnabled; IpProfile_SBCMSRPOfferSetupRole; IpProfile_SBCMSRPReinviteUpdateSupport; IpProfile_DataDiffServ; CpMediaRealm_TCPPortRangeStart; CpMediaRealm_TCPPortRangeEnd
42037	Updated to Version 7.20A.254.375 ■ Updated sections: Syslog Message Format; Configuring Message Session Relay Protocol (MSRP Empty Message Format parameter); Configuring Web Session and Access Settings (Password Change Interval updated); Configuring SIP Response Codes for Alternative Routing Reasons (18x followed by failure response); Obtaining License Key for Initial Activation (removed) ■ Updated parameters: WebUserPassChangeInterval (changed to WebPassChangeInterval and description); TargetOfChannel_ HotLineToneDuration (CLI command typo); QSIGTunnelingMode (CLI command added); SelectSourceHeaderForCalledNumber (Web name added and value typos); IpProfile_SBCGenerateNoOp (Web name changed); IpProfile_SBCRemoteUpdateSupport (Web name changed); IPProfile_LocalRingbackTone (Web name typo)

LTRT	Description
	■ New parameters: IPProfile_SBCMSRPEmpMsg
42038	■ Updated to Version 7.20A.254.475 ■ Updated sections: CDR Field Description (CALL_CONNECT); Saving and Loading ini Configuration File (file contents); Adding ELINs to the Location Information Server (ELIN tag); Disconnecting and Reconnecting HA; Direct Media or Media Bypass (transfer termination note); Configuring IP Groups (max) ■ New parameters: IpProfile_SBCMultipleCoders; FailedOptionsRetryTime; MinWebPasswordLen; IPGroup_TeamsMediaOptimization; IPGroup_ InternalMediaRealm; FormatDestPhoneNumber; SBC100TryingUponReinvite; TLSContexts_TlsRenegotiation; SBC100TryingUponReinvite ■ Updated parameters: DialPlanRule_Prefix (max digits); Answer Detector parameters removed (EnableAnswerDetector, AnswerDetectorActivityDelay, AnswerDetectorSilenceTime, AnswerDetectorRedirection, AnswerDetectorSensitivity); SBCSessionExpires (Web name updated); UseGatewayNameForOptions; SIPGatewayName; EtherGroupTable_Mode (typo values 0 and 1)
42039	Updated to Version 7.20A.256.024. ■ New sections: Configuring Push Notification Service; Configuring Push Notification Servers; Remote Monitoring of Device behind NAT ■ Updated sections: Customizing Access Levels per Web Page (note for pages can't be customized); Disabling Guarantee of DSPs on SDP Offer; Centralized Third-Party Routing Server (routing fallback); Accessing the Web Interface (remember username); CDR Field Description; Customizing CDRs for SBC Calls and Test Calls ■ New parameters: SystemLogSize; IPGroup_SIPSourceHostName; SIPTopologyHidingMode; ReserveDSPOnSDPOffer; PushNotificationServers; PNSReminderPeriod; PNSRegisterTimeout; IPGroup_TeamsMOInitialBehavior; FailedOptionsRetryTime; RemoteMonitoringEnable; RemoteMonitoringPeriod; RemoteMonitoringDeviceEnable; RemoteMonitoringAlarmsEnable; RemoteMonitoringPMEnable; RemoteMonitoringSIPUsersEnable ■ Updated parameters: FailedOptionsRetryTime; ResetWebPassword; SBCAdmissionRule_MaxBurst (typo); DNSRebindingProtectionEnabled (Web name); ShortCallSeconds (Web name); IPGroup_ TopologyHidingHeaderList (removed); DialPlanRule_Prefix (max); DialPlanRule_Tag (max); HTTPRemoteServices_HTTPType (Remote

LTRT	Description
	Monitoring); ProxySet_EnableProxyKeepAlive (Using OPTIONS on Active Server)
42140	Updated to Version 7.20A.256.366. ■ New sections: Packet Loss Indication in Syslog; Flex License Model; Viewing Flex License Utilization and Status Configuring WebSocket Tunnel with OVOC ■ Updated sections: Replacing the Corporate Logo with Text (CMD Shell removed); License Key (Flex License); V.150.1 Modem Relay (removed); Simultaneous Negotiation of Fax (T.38) and Modem (V.150.1) Relay (removed) ■ New parameters: PLThresholdLevelsPerMille; ■ Updated parameters: WSTunServer; WSTunServerPath; WSTunUsername; WSTunPassword; WSTunSecured; WSTunVerifyPeer; V.150.1 parameters removed (V1501AllocationProfile, V1501SSEPayloadTypeRx, V1501SSERedundancyDepth, V1501SPRTTransportChannel0MaxPayloadSize, V1501SPRTTransportChannel2MaxPayloadSize, V1501SPRTTransportChannel2MaxWindowSize, V1501SPRTTransportChannel3MaxPayloadSize)
42141	■ Updated sections: Configuring User-Defined Performance Monitoring MIBs (CLI added); Web Login Authentication using Smart Cards (Web parameter added); Fixed License Pool Model; Floating License Model (example); Viewing Syslog Messages (Syslog Viewer added); Viewing Device Status on Monitor Page (stats screenshot) ■ Updated parameters: EnableMgmtTwoFactorAuthentication (Web name); WSTunServerPath (default); WSTunUsername (default); WSTunPassword (default)
42142	Updated to Version 7.20A.256.713. ■ New sections: Hiding Caller and Callee CDR Field Values. ■ Updated sections: Coder Transcoding (MediaChannels note removed); Fax Fallback (removed); Generating Private Keys for TLS Contexts (private key size values); Viewing Device Status on Monitor Page (acPMSBCRegisteredUsersTable); Configuring an HTTP-based OVOC Service (ini names and CLI changed); Configuring Secured (HTTPS) Web (default OAMP and Additional Management Interfaces); Configuring TLS Certificate Contexts (ID 0 for all; default OAMP only 0); Configuring User-Defined Performance Monitoring MIBs (removed); Microsoft Teams with

LTRT	Description
	Local Media Optimization (name changed); removed all "SSL" and replaced with "TLS"; . ■ Updated parameters: SCTPHeartbeatInterval (ranges); SCTPInitialRTO (range); SCTPMinimumRTO (range); SCTPMaximumRTO (range); SCTPMaxPathRetransmit (range); SCTPMaxAssociationRetransmit (range); SCTPMaxDataTxBurst (range); SCTPMaxDataChunksBeforeSACK (range); SCTPTimeoutBeforeSACK (range); TLSContexts_TLSVersion (values added); TLSContexts_ServerCipherString (default); TLSContexts_ ClientCipherString (default); TLSContexts_DHKeySize (3072 added and default); MediaChannels; IpProfile_SBCEnhancedPlc (description); LdapConfiguration_LdapConfPassword (typo); LdapConfiguration_ LdapConfBindDn (typo); UserDefinedFailurePM (removed); SBCUserRegistrationGraceTime (max); FakeTCPalias (RFC added/web param removed). ■ New parameters: TLSContexts_KeyExchangeGroups; TLSContexts_ ServerCipherTLS13String; TLSContexts_ClientCipherTLS13String; CDRHistoryPrivacy
42143	Updated to Ver. 7.20A.256.721. ■ Updated sections: Configuring WebRTC (DTLS enabled); Configuring WebRTC (widget added); Generating Private Keys for TLS Contexts (4096 removed) ■ Updated parameters: IPGroup_TeamsMediaOptimization (Web/CLI/INI name changed); IPGroup_TeamsMOInitialBehavior (Web/CLI/INI name changed); TLSContexts_DHKeySize (4096 removed)
42144	Updated to Ver. 7.20A.258.006. ■ New sections: TLS Context Parameters Relevancy per Application ■ Updated sections: CDR Field Description Voice AI Connector ID/820, Voice AI Connector Name/821); Customizing CDRs for SBC Calls and Test Calls Voice AI Connector ID/820, Voice AI Connector Name/821); Restoring Default Corporate Logo (removed); Customizing the Favicon (restoring default removed); Configuring RTP Base UDP Port (sentence relating to old equation removed); Viewing Device Status on Monitor Page (typo with PM_gwINVITEDialogs); Direct Media or Media Bypass (note for ports) ■ Updated parameters: WSTunServerPath (default); InterfaceTable_ IPAddress (description); SIPInterface_TCPPort / SIPInterface_TLSPort (different ports); SIPTCPTimeout (max) ■ New parameters: ContextEngineID; StaticRouteTable_

LTRT	Description
	PreferredSourceInterfaceName
42145	Updated to Ver. 7.20A.260.005. ■ New sections: On-Demand SIPRec Sessions; Example of Call Variables for CDR Customization ■ Updated sections: CDR Field Description (Var Call User Defined added); Direct Media or Media Bypass (features not supported updated); Configuring HTTP POST Web Service (correction); Configuring an HTTP GET Web Service (correction); Collecting Debug Recording Messages (Wireshark improvements); Customizing CDRs for SBC Calls and Test Calls (Var Call User Defined and RADIUS CDR note); Configuring RADIUS Accounting (acct-status-type and acct-session-id); Third-Party Routing Server or AudioCodes Routing Manager (ARM Authentication) ■ Updated parameters: DialPlanRule_Prefix (case-sensitive); DialPlanRule_Tag (no spaces); SIPInterface_CallSetupRulesSetId (note for CSR with HTTP GET); IPGroup_TypeSBCServerAuthType (ARM Authentication option added)
42146	■ Updated to Ver. 7.20A.258.119. ■ Updated sections: CDR Field Description (Call Start for Var Call User Defined); Configuring Media (SRTP) Security ('a-crypto'); Coder Transcoding (CLI command for transcoding sessions); Configuring WebSocket Tunnel with OVOC (AWS only); Configuring Firewall Allowed Rules (removed HA firewall for Maintenance) ■ Updated parameters: AMDTimeout (CLI command); SBCAltRoutingReasonsList_ReleaseCauseCode (805 Admission Failure removed); TLSContexts_DHKeySize (4096 added); SBCRemoveSIPSFromNonSecuredTransport (web and CLI added); ENABLERAI (only Gateway), RAIHIGHTHRESHOLD (only Gateway), RAILOWTHRESHOLD (only Gateway); HAUnitIdName (sent to OVOC for traps) ■ New parameters: WSTunServerPath (description); WSTunUsername (description); WSTunPassword (description); WSTunVerifyPeer (description)
42149	■ Updated to Ver. 7.20A.258.246, 7.20A.258.271; 7.20A.260.095 ■ New sections: Playing Tone upon Call Connect ■ Updated sections: Configuring an HTTP GET Web Service; Configuring Classification Based on Tags (synchronous queries); Redirect Number and Calling Name (Display); (NTT/KOR added); Configuring IP Profiles (effect

LTRT	Description
	of global); Configuring a Routing Response Timeout (HTTP GET added); Configuring AD-Based Routing Rules (path to page); Viewing Device Status on Monitor Page (GUI LED for HA); Assigning Externally Created Private Keys to TLS Contexts (note matching private key and certificate); CRP sections removed ■ Updated parameters: HTTPSOnly (also REST); HTTPRemoteServices_ TimeOut (HTTP GET added); TransparentCoderOnDataCall (web name); IPGroup_SourceUriInput (description); MinSE (CLI command path); DialPlanRule_Prefix (wildcard wscape); Configuring IP Groups (note to not use IP Group #0) ■ New parameters: TLSIncrRootFileUrl; ENUMAllowNonDigits; ReceiveMultipleDTMFMethods; AutoUpdateFreqencySeconds (replaced AutoUpdateFrequency); PlayToneOnConnectFailureBehavior
42150	■ New sections: Notations and Priority Matching for Dial Plan Patterns ■ Updated sections: Configuring WebRTC; Debugging PSTN Calls through CLI (typo reset); Configuring Management User Accounts (typo); Configuring Upstream Hosts (note); Replacing Corporate Logo with Text ■ New parameters: IPGroup_TeamsDirectRoutingMode; CpMediaRealm_ UsedByRoutingServer; IPGroup_TeamsDirectRoutingMode; PreserveMultipartContentType; DTLSTimeBetweenTransmissions; SBCRenumberMID ■ Updated parameters: HTTPProxySyslogDebugLevel (description); LogoFileName (description); UseWebLogo (description)
42151	■ Updated to LTS Ver. 7.20A.258.457 (7.2.258-4) and LR Ver.7.20A.260.180 (7.2.260-2) ■ Updated sections: Configuring SNMP Community Strings (different read-only and read-write); Saving Configuration (traffic disruption removed); Configuring HTTP Proxy Servers (max rows); Configuring HTTP Locations (max. rows) ■ New parameters: SBCTerminateOptions ■ Updated parameters: EnableDiagnostics (removed from linux); Account_ RegistrarSearchMode (option 1 note); SBCCDRFormat_FieldType (Is Recorded added)
42154	■ Updated to LTS Ver. 7.20A.258.661 (7.2.258-6) ■ New sections: Configuring Web Service for Automatic Provisioning ■ Updated sections: Configuring Registration Accounts (served-serving

LTRT	Description
	combination); Configuring Debug Recording (note re OAMP); SIP-based Media Recording (typo); Configuring IP-to-IP Outbound Manipulations (headers); Configuring IP-to-IP Inbound Manipulations (headers); Configuring IP Network Interfaces (def. gateway); Configuring Static IP Routes (note); Configuring SIP Interfaces (TCP/TLS source ports dynamic); Interworking SIP Early Media (drawing updated); Configuring HTTP Locations (capacity); Notations and Priority Matching for Dial Plan Patterns (best match); Configuring Malicious Signatures (max. 20) ■ New parameters: IgnoreAuthorizationStale; ProxySet_IsProxyHotSwap (disable); SBCAlertTimeout; DateHeaderTimeSync; DateHeaderTimeSyncInterval; ProxySet_AcceptDHCPProxyList; SBCTerminateOPTIONS ■ Updated parameters: LdapConfServerMaxRespondTime (note); IncrementalIniFileURL (no reset); IPProfile_SBCSessionExpiresMode (description); SbcDtlsMtu (default); SyslogLogLevel(note)
42157	■ Updated to LTS Ver. 7.20A.258.750 (7.2.258-7) ■ New sections: Synchronizing Multiple SIP Accounts per IMS Specification ■ Updated sections: Configuring Push Notification Servers (typo); On-Demand SIPRec Sessions (note re INFO); Notations and Priority Matching for Dial Plan Patterns; G.711 Fax and Modem Transport Mode (typo gpmd); Filtering IP Network Traces using Wireshark-Like Expressions (example) ■ New parameters: SyncIMSAccounts; AUPDResetURLOnWebConfig ■ Updated parameters: IPGroup_SIPSourceHostName (typo); CIDNotification; SBCAlertTimeout (description)
42264	■ Updated to LTS Ver. 7.20A.258.920 (7.2.258-10.1) ■ New sections: Prerequisites for HA ■ Updated sections: Call Detail Records (syslog severity); Configuring WebRTC (hyperlink); Viewing Voice Channel Information; Configuring Dual Registration for SIP Entity (typo IP) ■ New parameters: IpProfile_SBCAllowOnlyNegotiatedPT ■ Updated parameters: IPGroup_SourceUriInput (note); SIPInterface_ UDPPort (port uniqueness); SIPInterface_TCPPort (port uniqueness); SIPInterface_TLSPort (port uniqueness); Using Dial Plan Tags for Routing Destinations (CSR); Using Dial Plan Tags for Call Setup Rules; IPProfile_ LocalRingbackTone (range); IPProfile_LocalHeldTone (range); EnableSIPREC (removed); DialPlanRule_Tag (valid chars);

LTRT	Description
	BrokenConnectionEventTimeout (def); ProxySet_IsProxyHotSwap (typo in enable); AudioCoders_Sce (value 2); AutoUpdatePredefinedTime; AutoUpdateFreqencySeconds; Account_ContactUser (note); IsUserPhone (CLI typo); IsUserPhoneInFrom (typo CLI)
42266	■ Updated sections: Call Setup Rule Examples (AD Teams example); AD-based Routing for Microsoft Teams or Skype for Business (CSR); Creating a Login Welcome Message (typo); Configuring Call Preemption for SBC Emergency Calls (note for resources); Notations and Priority Matching for Dial Plan Patterns (example typo); Downloading and Uploading a Configuration Package File (certificates); Digit Mapping ■ New parameters: AuthPassword (replaces Password) ■ Updated parameters: TLSContexts_ServerCipherString (OpenSSL URL); TLSContexts_ClientCipherString (OpenSSL URL); TLSContexts_ ServerCipherTLS13String (OpenSSL URL); TLSContexts_ ClientCipherTLS13String (OpenSSL URL); TLSContexts_DHKeySize (4096 removed); DefaultNumber (description); SIPTCPTimeout default); ReliableConnectionPersistentMode (description); Test_Call_ PlayToneIndex (range); IpProfile_SBCRemoteReferBehavior (description)
42271	■ Updated to LTS Ver. 7.20A.259.306 (7.2.258-16) ■ Updated sections: Configuring Logging Filter Rules (syslog and log type); Disabling Internal Switch Port for OSN (typo); Configuring WebSocket Tunnel with OVOC (cloud platforms); Configuring OVOC for QoE (no secondary address); Configuring Wireshark Packet Capturing using RPCAP (recommendations); Configuring WebRTC (hyperlink); Configuring Dual Registration for SIP Entity (typo IP) ■ New parameters: LdapConfiguration_VerifySubjectName; HeartBeatIntervalmsec; AccessList_Description ■ Updated parameters: HeldTimeout (description); DisableRS232 (default); EnableDID; MgmntLDAPGroups_Level (note); IP2IPRouting_DestType (description for all users); AccessList_Use_Specific_Interface (default); AccessList_PrefixLen (default)
14205	Initial document release.
42279	■ Updated sections: Notations and Priority Matching for Dial Plan Patterns (wildcards); DHCP-based Provisioning (mac); Configuring Management User Accounts (plain text passwords for shared settings); Configuring SNMP V3 Users (plain text passwords for shared settings); Configuring Table ini File Parameters ($$ removed)

LTRT	Description
	■ Updated parameters: CallSetupRules_AttributesToGet (max); IPGroup_ AuthenticationMode (description); WebUsers_SessionLimit (description); HostName (CLI path); ProxyIPListRefreshTime; HeartBeatIntervalmsec (description); Account_Password (note re question mark); InboundMediaLatchMode (description strict); EtherGroupTable_Mode (restrictions)
42343	■ Updated sections: Configuring WebRTC (Enforce Media Order); Saving and Loading the Configuration Package File (CLI Startup Script) ■ Updated parameters: IPGroup_SIPConnect (description); EnableSIPRemoteReset (description); SystemLogSize (range and default); SBCEnforceMediaOrder (Web parameter added)

Table of Contents

1 Introduction

This User's Manual describes how to configure and manage your AudioCodes Mediant 4000 SBC (hereafter, referred to as *device*).

Some features described in this *User's Manual* may only be supported by Latest Release (LR) software versions (i.e., 7.20A.254.xxx and later), and not by Long Term Support (LTS) software versions (i.e., 7.20A.204.xxx). For new features supported per software version type, refer to the relevant Release Note:

- **LTS Versions:** SBC-Gateway Series Release Notes for Long Term Support Versions 7.2
- **LR Versions:** SBC-Gateway Series Release Notes for Latest Release Versions 7.2

Product Overview

Mediant 4000 Session Border Controller (hereafter referred to as *device*) is a mid-to-high scale capacity member of AudioCodes field-proven hardware-based SBC product family, designed to offer enterprises and service providers a reliable and scalable SBC solution. The supports wide-ranging SIP interoperability, delivering service assurance and enabling scalable, reliable and secured connectivity between different VoIP networks.

The device provides a perfect solution for enterprises and large organizations such as contact centers, large data centers, hosted service providers and government institutions where security, reliability and high performance are critical.

The device includes comprehensive media security and SIP normalization capabilities. It offers full interoperability with an extensive list of IP-PBXs, unified communications solutions and SIP trunking provider networks.

The device provides robust protection for the IP communications infrastructure, preventing fraud and service theft and guarding against cyber-attacks and other service impacting events.

The device offers active/standby high availability and maintains high voice quality to deliver reliable enterprise VoIP communications. Advanced call routing mechanisms, network voice quality monitoring and branch survivability capabilities result in minimum communications downtime.

The device can be used for the following applications:

- SIP trunking
- Hosted PBX & UC as a Service
- IP contact centers
- Remote and mobile worker support
- SIP mediation between UC and IP-PBX systems
- Residential VoIP

AudioCodes MediaPack 5xx series (MP-504, MP-508, MP-524, and MP-532) of analog VoIP gateways provide service providers and enterprises with superior voice technology for connecting legacy telephones, fax machines and PBX systems with IP telephony networks, IP-PBX systems and unified communications solutions.

MediaPack 5xx gateways offer comprehensive SIP interoperability with leading vendors, delivering secure and reliable communications for enterprises and branch office locations.

Typographical Conventions

This document uses the following typographical conventions to convey information:

Table 1-1: Typographical Conventions

Convention	Description	Example
Text enclosed by a single quotation mark ('...')	Indicates Web interface parameters.	From the 'Debug Level' drop-down list, select **Basic**.
Boldface font	Indicates one of the following Web-based management interface elements: ■ A button ■ A selectable value ■ The navigational path to a Web page	Click the **Add** button.
Text enclosed by double quotation marks ("...")	Indicates values that you need to enter (type) in the Web interface.	In the 'IP Address' field, enter "10.10.1.1".
Courier font	Indicates CLI commands or ini-based file configuration.	At the CLI prompt, type the following: `# configure system`
Text enclosed by square brackets [...]	Indicates ini file parameters and values.	Configure the [GWDebugLevel] parameter to [1].
	Indicates a note bulletin providing important or useful information.	-
	Indicates a warning bulletin alerting you to potentially serious problems if a specific action is not taken.	-

Getting Familiar with Configuration Concepts and Terminology

Before using your device, it is recommended that you familiarize yourself with the basic configuration concepts and terminology. An understanding of the basic concepts and terminology will help you configure and manage your device more effectively and easily.

SBC Application

The objective of your configuration is to enable the device to forward calls between telephony endpoints in the SIP-based Voice-over-IP (VoIP) network. The endpoints (SIP entities) can be servers such as SIP proxy servers and IP PBXs, or end users such as IP phones. In the SIP world, the endpoints are referred to as SIP user agents (UA). The UA that initiates the call is referred to as the user agent client (UAC); the UA that accepts the call is referred to as the user-agent server (UAS).

The following table describes the main configuration concepts and terminology.

Table 1-2: Configuration Concepts and Terminology

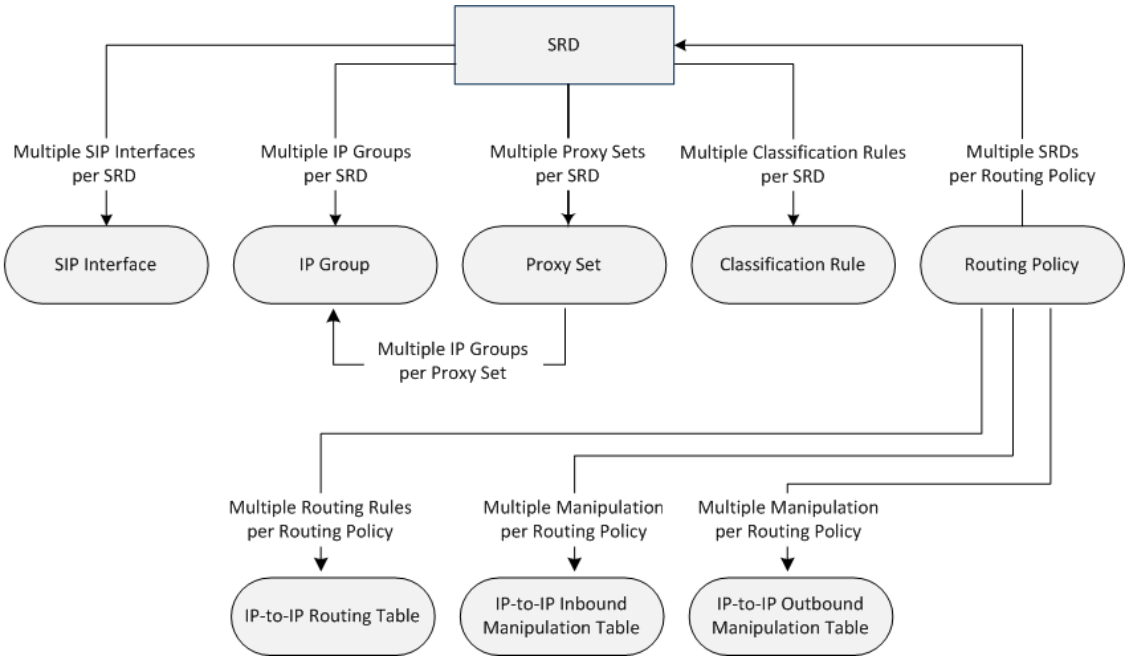
Configuration Terms	Description
IP Group	The IP Group is a logical representation of the SIP entity (UA) with which the device receives and sends calls. The SIP entity can be a server (e.g., IP PBX or SIP Trunk) or it can be a group of users (e.g., LAN IP phones). For servers, the IP Group is typically used to define the address of the entity (by its associated Proxy Set). IP Groups are used in IP-to-IP routing rules to denote the source and destination of the call.
Proxy Set	The Proxy Set defines the actual address (IP address or FQDN) of SIP entities that are servers (e.g., IP PBX). As the IP Group represents the SIP entity, to associate an address with the SIP entity, the Proxy Set is assigned to the IP Group. You can assign the same Proxy Set to multiple IP Groups (belonging to the same SRD).
SIP Interface	The SIP Interface represents a Layer-3 network. It defines a local listening port for SIP signaling traffic on a local, logical IP network interface. The term *local* implies that it's a logical port and network interface on the device. The SIP Interface is used to receive and send SIP messages with a specific SIP entity (IP Group). Therefore, you can create a SIP Interface for each SIP entity in the VoIP network with which your device needs to communicate. For example, if your VoIP network consists of three SIP entities -- a SIP Trunk, a LAN IP PBX, and remote WAN users -- a SIP Interface can be created for each of these Layer-3 networks. The SIP Interface is associated with the SIP entity, by assigning it to an SRD that is in turn, assigned to the IP Group of the SIP entity.

Configuration Terms	Description
Media Realm	The Media Realm defines a local UDP port range for RTP (media) traffic on any one of the device's logical IP network interfaces. The Media Realm is used to receive and send media traffic with a specific SIP entity (IP Group). The Media Realm can be associated with the SIP entity, by assigning the Media Realm to the IP Group of the SIP entity, or by assigning it to the SIP Interface associated with the SIP entity.
SRD	The SRD is a logical representation of your entire SIP-based VoIP network (Layer 5) containing groups of SIP users and servers. The SRD is in effect, the foundation of your configuration to which all other previously mentioned configuration entities are associated. For example, if your VoIP network consists of three SIP entities -- a SIP Trunk, a LAN IP PBX, and remote WAN users -- the three SIP Interfaces defining these Layer-3 networks would all assigned to the same SRD. Typically, only a single SRD is required and this is the recommended configuration topology. As the device provides a default SRD, in a single SRD topology, the device automatically assigns the SRD to newly created configuration entities. Thus, in such scenarios, there is no need to get involved with SRD configuration. Multiple SRDs are required only for multi-tenant deployments, where it "splits" the device into multiple logical devices. For multiple SRDs, the SRD can be configured with a Sharing Policy. The Sharing Policy simply means whether the SRD's resources (SIP Interfaces, IP Groups, and Proxy Sets) can be used by other SRDs. For example, if all tenants route calls with the same SIP Trunking service provider, the SRD of the SIP Trunk would be configured as a *Shared* Sharing Policy. SRDs whose resources are not shared, would be configured with an *Isolated* Sharing Policy.
IP Profile	The IP Profile is an optional configuration entity that defines a wide range of call settings for a specific SIP entity (IP Group). The IP Profile includes signaling and media related settings, for example, jitter buffer, silence suppression, voice coders, fax signaling method, SIP header support (local termination if not supported), and media security method. The IP Profile is in effect, the interoperability "machine" of the device, enabling communication between SIP endpoints that "speak" different call "languages". The IP Profile is associated with the SIP entity, by assigning the IP Profile to the IP Group of the SIP entity.
Classification	Classification is the process that identifies the incoming call (SIP dialog

Configuration Terms	Description
	request) as belonging to a specific SIP entity (IP Group). There are three chronological classification stages, where each stage is done only if the previous stage fails. The device first attempts to classify the SIP dialog by checking if it belongs to a user that is already registered in the device's registration database. If this stage fails, the device checks if the source IP address is defined for a Proxy Set and if yes, it classifies it to the IP Group associated with the Proxy Set. If this fails, the device classifies the SIP dialog using the Classification table, which defines various characteristics of the incoming dialog that if matched, classifies the call to a specific IP Group. The main characteristics of the incoming call is the SIP Interface that is associated with the SRD for which the Classification rule is configured.
IP-to-IP Routing	IP-to-IP routing rules define the routes for routing calls between SIP entities. As the SIP entities are represented by IP Groups, the routing rules typically employ IP Groups to denote the source and destination of the call. For example, to route calls from the IP PBX to the SIP Trunk, the routing rule can be configured with the IP PBX as the source IP Group and the SIP Trunk as the destination IP Group. Instead of IP Groups, various other source and destination methods can be used. For example, the source can be a source host name while the destination can be an IP address or based on an LDAP query.
Inbound and Outbound Manipulation	Inbound and Outbound Manipulation lets you manipulate the user part of the SIP URI in the SIP message for a specific entity (IP Group). Inbound manipulation is done on messages received from the SIP entity; outbound manipulation is done on messages sent to the SIP entity. Inbound manipulation lets you manipulate the user part of the SIP URI for source (e.g., in the SIP From header) and destination (e.g., in the Request-URI line) in the incoming SIP dialog request. Outbound manipulation lets you manipulate the user part of the Request-URI for source (e.g., in the SIP From header) or destination (e.g., in the SIP To header) or calling name, in outbound SIP dialog requests. The Inbound and Outbound manipulation are associated with the SIP entity, by configuring the rules with incoming characteristics such as source IP Group and destination host name. The manipulation rules are also assigned a Routing Policy, which in turn, is assigned to IP-to-IP routing rules. As most deployments require only one Routing Policy, the default Routing Policy is automatically assigned to the manipulation rules and to the routing rules.

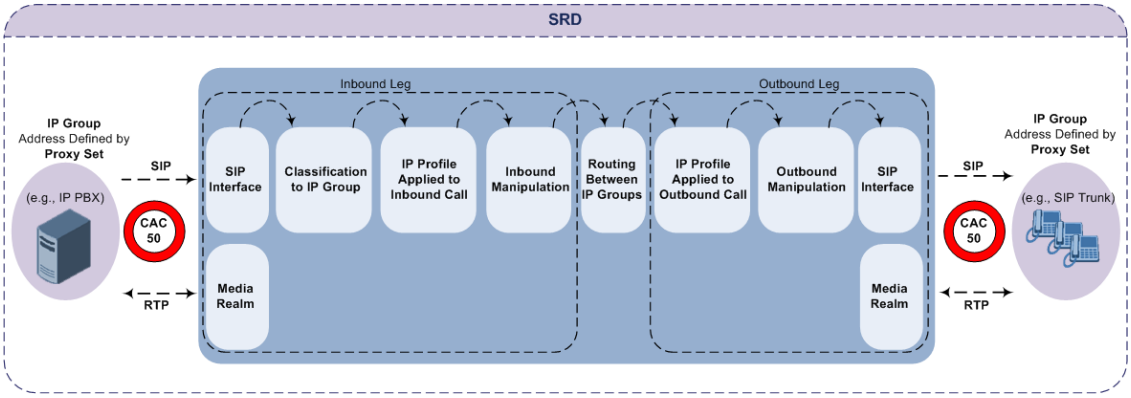
Configuration Terms	Description
Routing Policy	Routing Policy logically groups routing and manipulation (inbound and outbound) rules to a specific SRD. It also enables Least Cost Routing (LCR) for routing rules and associates an LDAP server for LDAP-based routing. However, as multiple Routing Policies are required only for multi-tenant deployments, for most deployments only a single Routing Policy is required. When only a single Routing Policy is required, handling of this configuration entity is not required as a default Routing Policy is provided, which is automatically associated with all relevant configuration entities.
Call Admission Control	Call Admission Control (CAC) lets you configure the maximum number of permitted concurrent calls (SIP dialogs) per IP Group, SIP Interface, SRD, or user.
Accounts	Accounts are used to register or authenticate a "served" SIP entity (e.g., IP PBX) with a "serving" SIP entity (e.g., a registrar or proxy server). The device does this on behalf of the "served" IP Group. Authentication (SIP 401) is typically relevant for INVITE messages forwarded by the device to a "serving" IP Group. Registration is for REGISTER messages, which are initiated by the device on behalf of the "serving" SIP entity.

The associations between the configuration entities are summarized in the following figure:



The main configuration entities and their involvement in the call processing is summarized in following figure. The figure is used only as an example to provide basic understanding of the

configuration terminology. Depending on configuration and network topology, the call process may include additional stages or a different order of stages.



1. The device determines the SIP Interface on which the incoming SIP dialog is received and thus, determines its associated SRD.
2. The device classifies the dialog to an IP Group (origin of dialog), using a specific Classification rule that is associated with the dialog's SRD and that matches the incoming characteristics of the incoming dialog defined for the rule.
3. IP Profile and inbound manipulation can be applied to incoming dialog.
4. The device routes the dialog to an IP Group (destination), using the IP-to-IP Routing table. The destination SRD (and thus, SIP Interface and Media Realm) is the one assigned to the IP Group. Outbound manipulation can be applied to the outgoing dialog.

Part I

Getting Started with Initial Connectivity

2 Introduction

This part describes how to initially access the device's management interface and change its default IP address to correspond with your networking scheme.

3 Default IP Address

The device is shipped with a factory default networking address for operations, administration, maintenance, and provisioning (OAMP), through its LAN interface, as shown in the table below. You can use this address to initially access the device from any of its management tools (embedded Web server, OVOC, or Telnet/SSH). You can also access the device through the console CLI, by connecting the device's serial (RS-232) port to a PC.

Table 3-1: Default VoIP LAN IP Address for OAMP

IP Address	Value
IP Address	192.168.0.2 (assigned to the first Ethernet Port Group - top-left ports 1 and 2)
Prefix Length	24 (255.255.255.0)
Default Gateway	0.0.0.0
VLAN	1
Ethernet Group / Port	GROUP_1 / GE_1 or GE_2

4 Configuring VoIP LAN Interface for OAMP

You can change the device's default OAMP IP address, using any of the following methods:

- Embedded HTTP/S-based Web server (see Web Interface)
- Embedded CLI (see CLI)
- ini file (see Changing OAMP Address through ini File on page 15

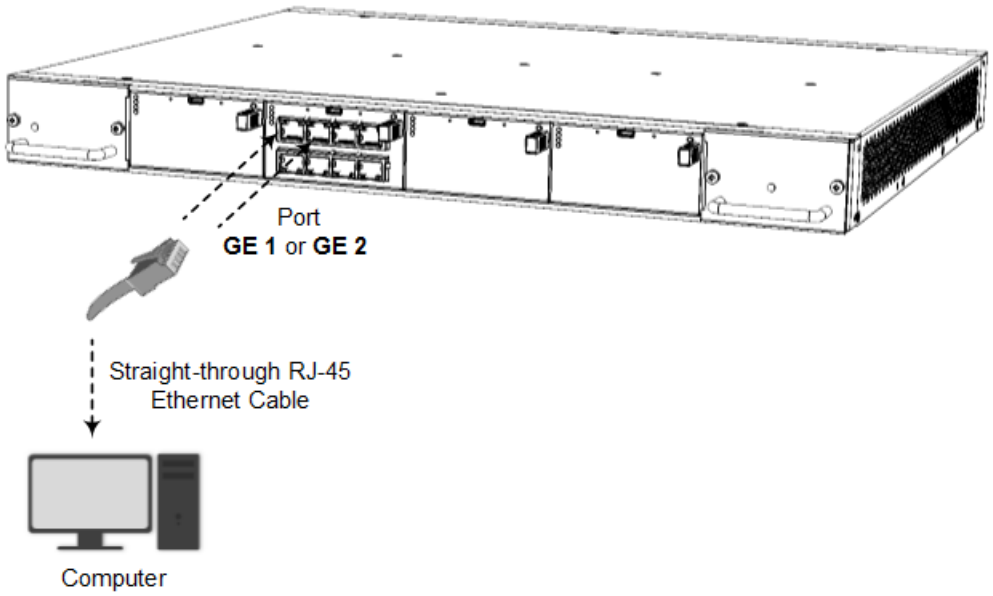
> If you are implementing the High Availability feature, see also HA Overview for initial setup.

Changing OAMP Address through Web Interface

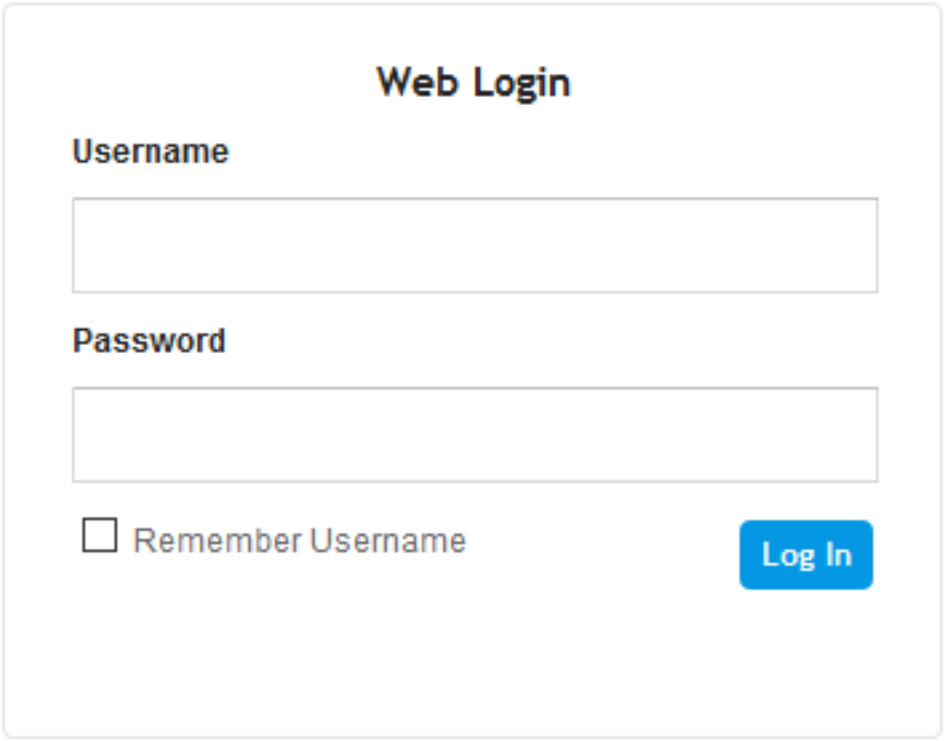
You can change the device's default OAMP networking address through the Web-based management tool (Web interface). The default IP address is used to initially access the device.

➢ **To change the default OAMP network address through Web interface:**

1. Connect one of the first two Ethernet ports (GE 1 or GE 2) on the top-left row of the SBC module, located on the front panel directly to the network interface of your computer, using a straight-through Ethernet cable.



2. Change the IP settings of your computer to correspond with the default OAMP IP address and subnet mask of the device.
3. Access the Web interface:
   a. On your computer, start a Web browser and in the URL address field, enter the default IP address of the device; the Web interface's Web Login screen appears:



b. In the 'Username' and 'Password' fields, enter the case-sensitive, default login username ("Admin") and password ("Admin").

c. Click **Log In**.

4. Configure the Ethernet port(s) that you want to use for the OAMP interface:

a. In the Ethernet Groups table, configure an Ethernet Group by assigning it up to two ports (two ports provide optional, port-pair redundancy). For more information, see Configuring Physical Ethernet Ports.

b. In the Physical Ports table, configure port settings such as speed and duplex mode (see Configuring Physical Ethernet Ports).

c. In the Ethernet Devices table, configure an Ethernet Device by assigning it the Ethernet Group and a VLAN ID (see Configuring Underlying Ethernet Devices).

5. Modify the OAMP interface address to suite your network environment:

a. Open the IP Interfaces table (see Configuring IP Network Interfaces).

b. Select the OAMP interface ("O+M+C"), and then click **Edit**.

c. From the 'Ethernet Device' drop-down list, select the Ethernet Device that you configured in the previous step.

d. Under the **IP Address** group, change the IP address to correspond with your network IP addressing scheme.

e. Under the **DNS** group, configure the DNS server, if required.

f. Click **Apply**; the new OAMP address is applied to the device and your connectivity to the device's Web interface at its previous OAMP address is now lost.

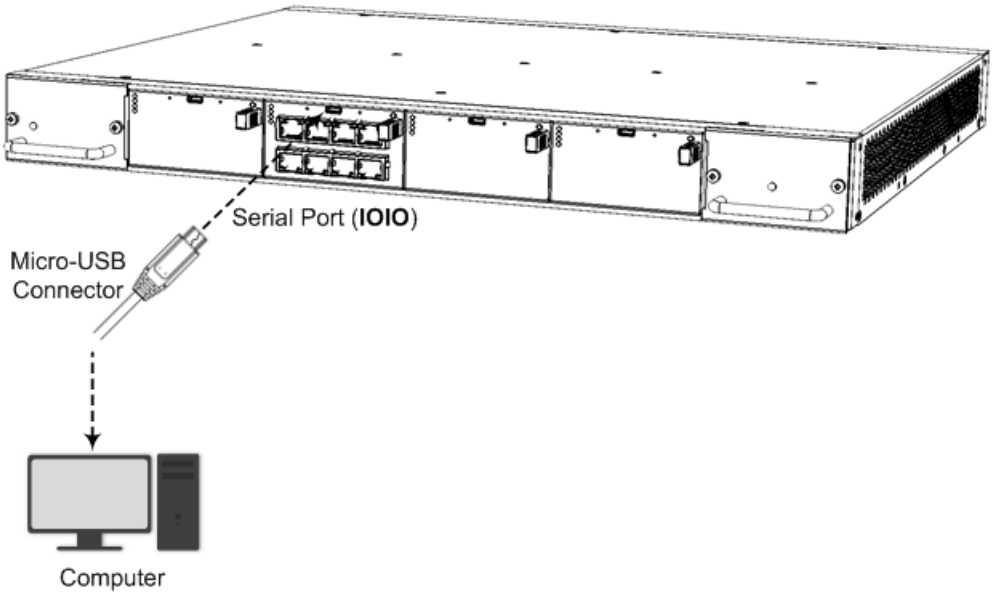
6. Change the IP settings of your computer to correspond with the new OAMP IP address and subnet mask that you assigned the device.
7. Access the device using the new OAMP IP address, and then on the Web interface's toolbar, click the **Save** button.
8. Re-cable the device to the desired network. Your can now access the device's management interfaces using the new OAMP address.

Changing OAMP Address through CLI

You can change the default OAMP IP address through the device's CLI. The procedure uses the regular CLI commands. Alternatively, you can use the CLI Wizard utility to set up your device with the initial OAMP settings. The utility provides a fast-and-easy method for initial configuration of the device through CLI. For more information, refer to the *CLI Wizard User's Guide*.

➢ **To configure the OAMP IP address through CLI:**

1. Connect the RS-232 port of the device to the serial communication port on your computer. For more information, refer to the Hardware Installation Manual.



2. Establish serial communication with the device using a terminal emulator program such as HyperTerminal, with the following communication port settings:
   - Baud Rate: 115,200 bps
   - Data Bits: 8
   - Parity: None
   - Stop Bits: 1
   - Flow Control: None
3. At the CLI prompt, type the username (default is "Admin" - case sensitive):

```
Username: Admin
```

4. At the prompt, type the password (default is "Admin" - case sensitive):

```
Password: Admin
```

5. At the prompt, type the following:

```
enable
```

6. At the prompt, type the password again:

```
Password: Admin
```

7. Access the Network configuration mode:

```
# configure network
```

8. Access the IP Interfaces table:

```
(config-network)# interface network-if 0
(network-if-0)#
```

9. Configure the IP address:

```
(network-if-0)# ip-address <IP address>
```

10. Configure the prefix length:

```
(network-if-0)# prefix-length <prefix length / subnet mask, e.g., 16>
```

11. Configure the Default Gateway address:

```
(network-if-0)# gateway <IP address>
```

12. Apply your settings:

```
(network-if-0)# activate
```

13. Cable the device to your network. You can now access the device's management interface using this new OAMP IP address.

Changing OAMP Address through ini File

You can change the device's default OAMP networking address by loading an ini file with the new OAMP address, through the device's Web interface.

➢ **To change the default OAMP network address through ini file:**

1. Create an ini file using a regular text-based editor program (such as Notepad) and save it with the file extension name **.ini** (e.g., myfile.ini).
2. Copy-and-paste the below ini file table parameter (InterfaceTable) into the ini file:

```
[ InterfaceTable ]
FORMAT InterfaceTable_Index = InterfaceTable_ApplicationTypes,
InterfaceTable_InterfaceMode, InterfaceTable_IPAddress, InterfaceTable_
PrefixLength, InterfaceTable_Gateway, InterfaceTable_InterfaceName,
InterfaceTable_PrimaryDNSServerIPAddress, InterfaceTable_
SecondaryDNSServerIPAddress, InterfaceTable_UnderlyingDevice;
InterfaceTable 0 = 6, 10, <IP Address>, <Prefix Length>, <Default Gateway>,
"O+M+C", 0.0.0.0, 0.0.0.0, "vlan 1";
[ \InterfaceTable ]
```

3. Modify the <IP Address>, <Prefix Length>, and <Default Gateway> parameters (but remove the angle brackets).
4. Save the file.
5. Connect to the device's Web interface using the default OAMP address (see Changing OAMP Address through Web Interface on page 11).
6. Open the Configuration File page (**Setup** menu > **Administration** tab > **Maintenance** folder > **Configuration File**).
7. Under the INI File group, click the **Browse** button to select the file, and then click **Load INI File**; the device loads the file and saves the configuration to flash memory with a device reset.
8. Disconnect the device from your PC and re-connect it to your network.
9. Access the device using the new OAMP interface address.

Part II

Management Tools

5 Introduction

This part describes the various management tools that you can use to configure the device:

- Embedded HTTP/S-based Web server - see Web-based Management
- Embedded Command Line Interface (CLI) - see CLI-Based Management
- Simple Network Management Protocol (SNMP) - see SNMP-Based Management
- Configuration *ini* file - see INI File-Based Management
- REST API - see REST-Based Management on page 101

- Some configuration settings can only be done using a specific management tool.
- For a list and description of all the configuration parameters, see Configuration Parameters Reference.

6 Web-Based Management

The device provides an embedded Web server (hereafter referred to as *Web interface*), supporting fault management, configuration, accounting, performance, and security (FCAPS). The Web interface provides a user-friendly, graphical user interface (GUI), which can be accessed using any standard Web browser. Access to the Web interface can be controlled by various security mechanisms such as login username and password, read-write privileges, and limiting access to specific IP addresses.

- The Web interface allows you to configure most of the device's settings. However, additional configuration parameters may exist that are not available in the Web interface and which can only be configured using other management tools.
- Some Web interface pages and parameters are available only for certain hardware configurations or software features. The software features are determined by the installed License Key (see License Key).

Getting Acquainted with the Web Interface

This section provides a description of the Web interface's graphical user interface (GUI).

Computer Requirements

The client computer accessing the device's Web interface requires the following prerequisites:

- A network connection to the device.
- One of the following Web browsers:
  - Microsoft™ Internet Explorer™ (Version 11.0.13 or later)
  - Mozilla Firefox® (Version 5.02 or later)
  - Google Chrome (Version 50 or later)

The Web browser must be JavaScript-enabled.

- Recommended screen resolutions: 1024 x 768 pixels, or 1280 x 1024 pixels.

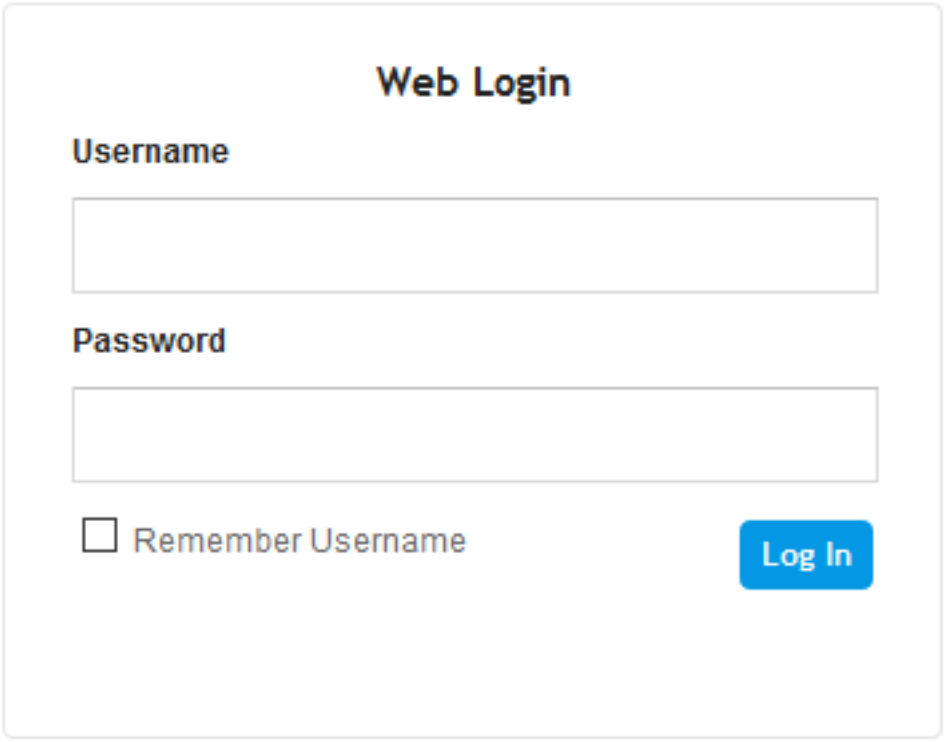
Accessing the Web Interface

The following procedure describes how to access the Web interface.

➢ **To access the Web interface:**

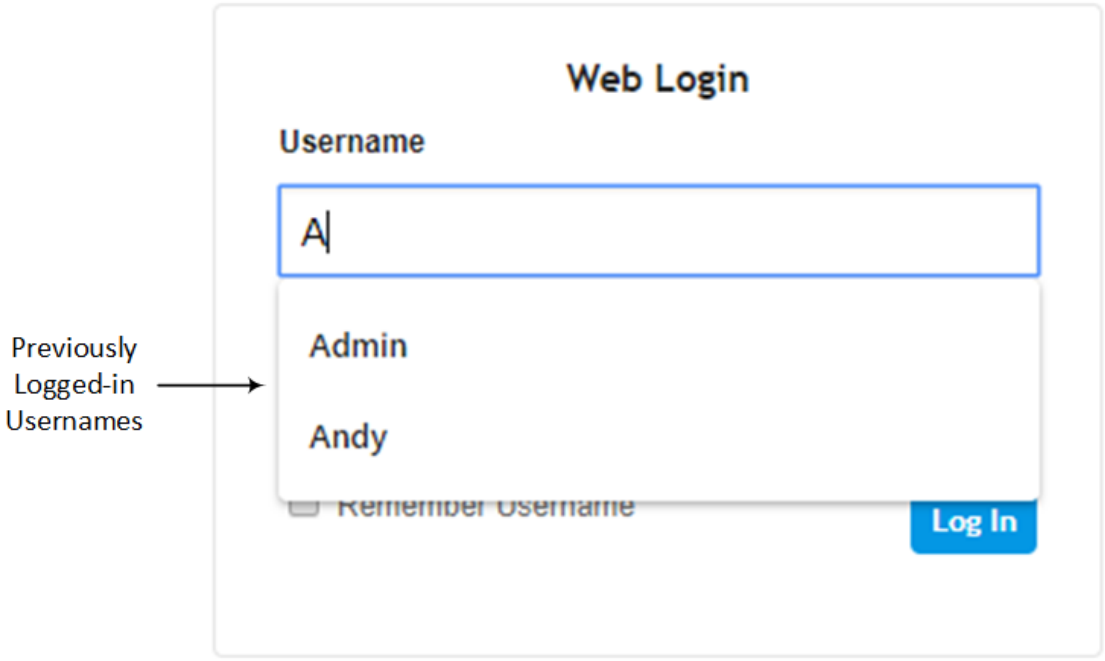
1. Open a standard Web browser.

2. In the Web browser, enter the device's OAMP IP address (e.g., http://10.1.10.10); the Web interface's Web Login window appears:



3. In the 'Username' and 'Password' fields, enter your username and password, respectively.
4. If you want the Web browser to remember your username for future logins, select the 'Remember Username' check box. On your next login attempt, the 'Username' field will be automatically populated with your username.
5. Click **Log In**.

By default, autocompletion of the login username is enabled, whereby the 'Username' field predicts the rest of the username while you are typing, by displaying a drop-down list with previously entered usernames, as shown in the example below. To disable autocompletion, use the [WebLoginBlockAutoComplete] ini file parameter.



- The default login username and password is **Admin** and **Admin**, respectively. To change the login credentials, see Configuring Management User Accounts.
- The username and password is case-sensitive.
- By default, Web access is only through the IP address of the OAMP interface. However, you can allow access from all of the device's IP network interfaces, by setting the EnableWebAccessFromAllInterfaces parameter to 1.
- Depending on your Web browser's settings, a security warning box may be displayed. The reason for this is that the device's certificate is not trusted by your PC. The browser may allow you to install the certificate, thus skipping the warning box the next time you connect to the device. If you are using Windows Internet Explorer, click **View Certificate**, and then **Install Certificate**. The browser also warns you if the host name used in the URL is not identical to the one listed in the certificate. To resolve this, add the IP address and host name (ACL_nnnnnn, where *nnnnnn* is the serial number of the device) to your hosts file, located at /etc/hosts on UNIX or C:\Windows\System32\Drivers\ETC\hosts on Windows; then use the host name in the URL (e.g., https://ACL_280152). Below is an example of a host file:
  127.0.0.1 localhost
  10.31.4.47 ACL_280152

Areas of the GUI

The areas of the Web interface's GUI are shown in the figure below and described in the subsequent table.

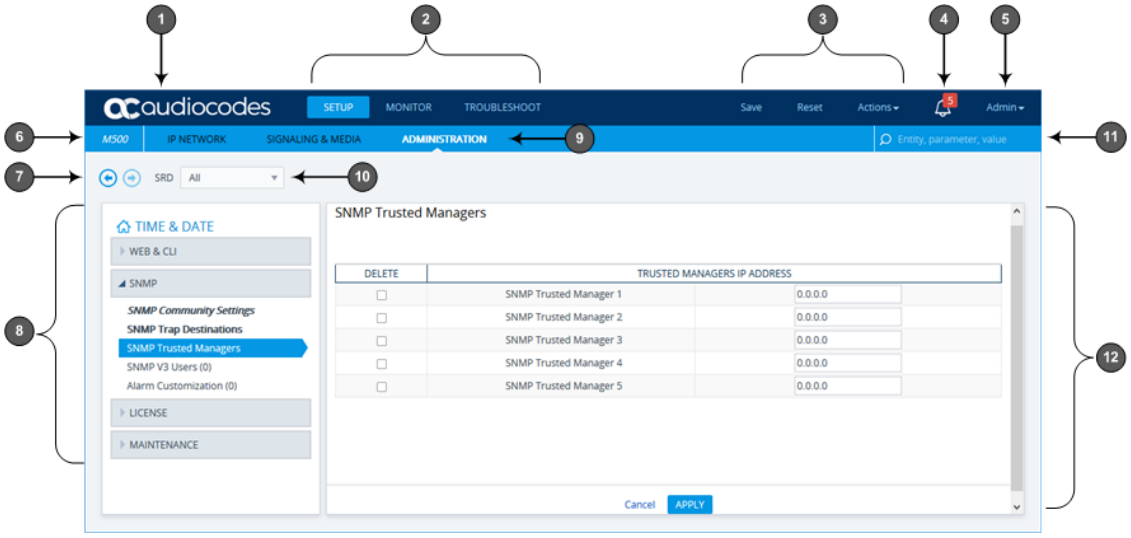
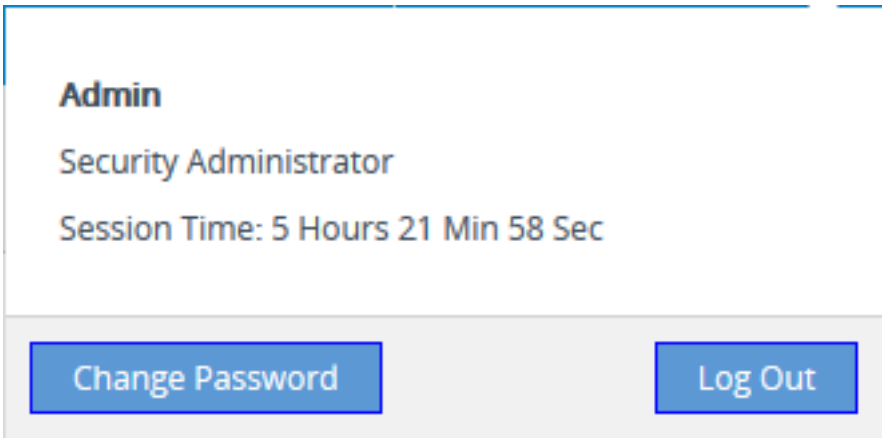


Table 6-1: Description of the Web GUI Areas

Item#	Description
1	Company logo. To customize the logo, see Replacing the Corporate Logo. If you click the logo, the Topology View page opens (see Building and Viewing SIP Entities in Topology View on page 499).
2	Menu bar containing the menus.
3	Toolbar providing frequently required command buttons. ■ **Save:** Saves configuration changes to the device's flash memory (without resetting the device). If you make a configuration change, the button is surrounded by a red-colored border as a reminder to save your settings to flash memory. ■ **Reset:** Opens the Maintenance Actions page, which is used for performing various maintenance procedures such as resetting the device (see Basic Maintenance). If you make a configuration change that takes effect only after a device reset, the button is surrounded by a red-colored border as a reminder; otherwise, your changes revert to previous settings if the device resets or powers off. ■ **Actions:** ✓ **Configuration File:** Opens the Configuration File page, which is used for saving the *ini* file to a folder on your PC, or for loading an ini file to the device (see Configuration File). ✓ **Auxiliary Files:** Opens the Auxiliary Files page, which is used for loading Auxiliary files to the device (see Loading Auxiliary Files through Web Interface). ✓ **License Key:** Opens the License Key page, which is used for installing a new License Key file (see Installing License Key through Web Interface).

Item#	Description
	✓ **Software Upgrade:** Starts the Software Upgrade Wizard for upgrading the device's software (see Software Upgrade). ✓ **Switchover:** Opens the High Availability Maintenance page, which is used for switching between Active and Redundant unit (see Initiating an HA Switchover). ✓ **Configuration Wizard:** Opens the SBC Configuration Wizard, which is used for quick-and-easy configuration of the device (see SBC Configuration Wizard).
4	Alarm bell icon displaying the number of active alarms generated by the device. The color of the number indicates the highest severity of an active alarm. If you click the icon, the Active Alarms table is displayed. If the device is in HA mode, the icon displays the number of currently active alarms raised by both Active and Redundant devices as well as the highest severity of these active alarms. For more information, see Viewing Active Alarms.
5	Button displaying the username of the currently logged in user. If you click the button, a drop-down box appears: ■ Displays information of the currently logged-in user (see Viewing Logged-In User Information) ■ **Change Password** button to change your login password (see Changing Login Password by All User Levels on page 63) ■ **Log Out** button to log out the Web session (see Logging Off the Web Interface) 
6	Product name of device. **Note:** ■ If you configure a hostname for the device (see Configuring a Hostname for the Device on page 109), the hostname is displayed instead of the product

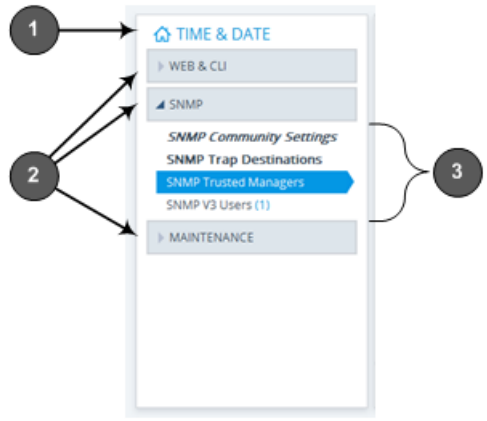
Item#	Description
	name. ■ You can customize the product name, as described in Customizing the Product Name on page 43.
7	Back and Forward buttons that enable quick-and-easy navigation through previously opened pages. This is especially useful when you find that you need to return to a previously accessed page, and then need to go back to the page you just left. ■ Back button: Goes back to the previously accessed page. ■ Forward button: Opens the page that you initially left using the back button. The button is available only if you have used the Back button.
8	Navigation pane displaying the Navigation tree containing the commands (items) for opening the configuration pages (see Navigation Tree).
9	Tab bar containing tabs pertaining to the selected menu: ■ Setup menu: ✓ IP Network tab ✓ Signaling & Media tab ✓ Administration tab ■ Monitor menu: Monitor tab ■ Troubleshoot menu: Troubleshoot tab
10	SRD filter. When your configuration includes multiple SRDs, you can filter tables in the Web interface by SRD. For more information, see Filtering Tables in Web Interface by SRD.
11	Search box for searching parameter names and values (see Searching for Configuration Parameters).
12	Work pane where configuration pages are displayed.

Accessing Configuration Pages from Navigation Tree

Accessing configuration pages is a three-fold process that consists of selecting a menu on the menu bar, a tab on the tab bar, and then a page item in the Navigation pane. The Navigation pane provides the Navigation tree, which is a tree-like structure of folders and page items that open configuration pages in the Work pane. The hierarchical structure and organization of the items in the Navigation tree allow you to easily drill-down and locate the required item.

The Navigation tree consists of the following areas:

- **Home** : (Callout #1) First ("home") page displayed when a menu-tab combination is initially selected. For example, the home page of the **Setup** menu - **Administration** tab combination is the Time & Date page.
- **Folders:** (Callout #2) Folders group items of similar functionality. To open and close a folder, simply click the folder name.
- **Items:** (Callout #3) Items open configuration pages. In some cases, an item may be listed under a sub-item. An item can open a page containing stand-alone parameters or a table. If it opens a page with stand-alone parameters, the item is displayed in italics. If it opens a page with a table, the item is displayed in regular font, or bold font to indicate an item that is commonly required.



The items of the Navigation tree depend on the menu-tab combination, selected from the menu bar and tab bar, respectively. The menus and their respective tabs are listed below:

- **Setup menu:**
  - **IP Network** tab
  - **Signaling & Media** tab
  - **Administration** tab
- **Monitor menu:** Monitor tab
- **Troubleshoot menu:** Troubleshoot tab

When you open the Navigation tree, folders containing commonly required items are opened by default, allowing quick access to their pages.

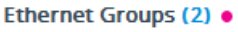
Items that open pages containing tables provide the following indications in the Navigation tree:

- **Number of configured rows**. For example, the item below indicates that two rows have been configured:

  Ethernet Groups (2)

  If you have filtered the Web interface display by SRD, the number reflects only the rows that are associated with the filtered SRD.

- **Invalid row configuration.** If you have configured a row with at least one invalid value, a red-colored icon is displayed next to the item, as shown in the following example:

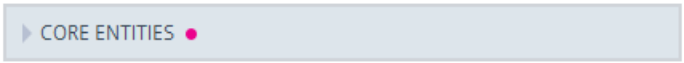
  If you hover your cursor over the icon, it displays the number of invalid rows (*lines*).

- **Association with an invalid row:** If you have associated a parameter of a row with a row of a different table that has an invalid configuration, the item appears with an arrow and a red-colored icon, as shown in the following example:

  If you hover your cursor over the icon, it displays the number of rows in the table that are associated with invalid rows.

- **Folder containing an item with an invalid row:** If a folder contains an item with an invalid row (or associated with an invalid row), the closed folder displays a red-colored icon, as shown in the following example:

  If you hover your cursor over the icon, it displays the names of the items that are configured with invalid values. If you have filtered the Web interface display by SRD, only items with invalid rows that are associated with the filtered SRD are displayed.

➢ To open a configuration page:

1. On the menu bar, click the required menu.
2. On the tab bar, click the required tab; the Navigation tree displays the items pertaining to the selected menu-tab combination.
3. In the Navigation pane, open the folder in which the required item is located. The folders are opened and closed by clicking the title of the folder. When opened, the folder's arrow is displayed as ◢ ; when closed, the arrow is displayed as ▶ .
4. In the folder, click the required item; the page is displayed in the Work pane.

You can also easily navigate through previously accessed pages, using the **Back** and **Forward** buttons located above the Navigation pane:

- **Back** button: Click to go back to the previously accessed page or keep on clicking until you reach any other previously accessed page.
- **Forward** button: Click to open the page that you just left as a result of clicking the **Back** button.

These buttons are especially useful when you find that you need to return to a previously accessed page, and then need to go back to the page you just left.

Depending on the access level (e.g., Monitor level) of your Web user account, certain pages may not be accessible or may be read-only (see Configuring Management User Accounts). For read-only privileges:

- Read-only pages with stand-alone parameters: "Read Only Mode" is displayed at the bottom of the page.
- Read-only pages with tables: Configuration buttons (e.g., **New** and **Edit**) are missing.

Configuring Stand-alone Parameters

Parameters that are not contained in a table are referred to as *stand-alone* parameters.

- If you change the value of a parameter (before clicking **Apply**), the parameter's field is highlighted, as shown in the example below:

- If you change the value of a parameter from its default value and then click **Apply**, a dot appears next to the parameter's field, as shown in the example below:

- If you change the value of a parameter that is displayed with a lightning-bolt icon (as shown in the example below), you must save your settings to flash memory with a device reset for your changes to take effect. When you change such a parameter and then click **Apply**, the **Reset** button on the toolbar is encircled by a red border. If you click the button, the Maintenance Actions page opens, which provides commands for doing this (see Basic Maintenance).

- Typically required parameters are displayed in bold font.
- If you enter an invalid value for a parameter and then click **Apply**, a message box appears notifying you of the invalid value. Click **OK** to close the message. The parameter reverts to its previous value and the field is surrounded by a colored border, as shown in the figure below:

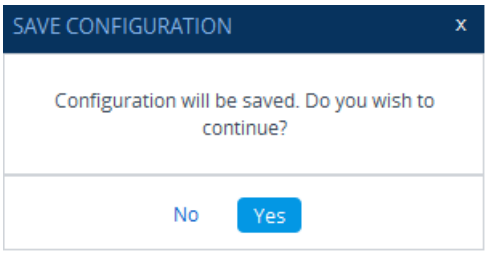
- To get help on a parameter, simply hover your mouse over the parameter's field and a pop-up help appears, displaying a brief description of the parameter.

The following procedure describes how to configure stand-alone parameters.

➢ **To configure a stand-alone parameter:**

1. Modify the parameter's value as desired.
2. Click **Apply**; the changes are saved to the device's volatile memory (RAM).

3. Save the changes to the device's non-volatile memory (flash):
   - If a device reset is not required:
     i. On the toolbar, click **Save**; a confirmation message box appears:



     ii. Click **Yes** to confirm; the changes are save to flash memory.
   - If a device reset is required:
     i. On the toolbar, click **Reset**; the Maintenance Actions page opens.
     ii. Click **Reset**; the device saves the changes to flash memory and then resets.

> When you click **Apply**, your changes are saved only to the device's volatile memory and thus, revert to their previous settings if the device later undergoes a hardware reset, a software reset (without saving to flash) or powers down. Therefore, make sure that you save your configuration to the device's flash memory.

Configuring Table Parameters

A typical configuration table is shown below and subsequently described:

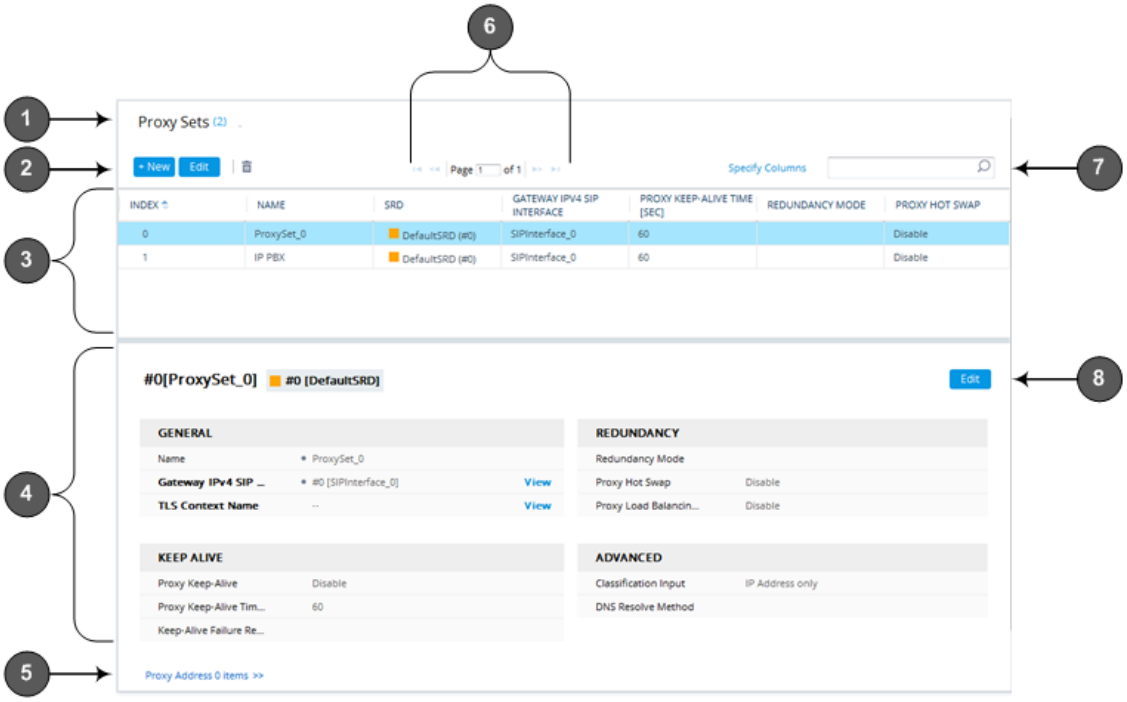


Table 6-2: General Description of Configuration Tables

Item#	Button	Description
1	-	Page title (i.e., name of table). The page title also displays the number of configured rows as well as the number of invalid rows. For more information on invalid rows, see Invalid Value Indications.
2	+ New	Adds a new row to the table (see Adding Table Rows).
	Edit	Modifies the selected row (see Modifying Table Rows).
	Clone	Adds a new row with similar settings as the selected row (i.e., clones the row). For more information, see Cloning SRDs. **Note:** The button appears only in the SRDs table.
		Deletes the selected row (see Deleting Table Rows).
		Changes the index position of a selected row (see Changing Index Position of Table Rows).
	Action	Drop-down menu providing commands (e.g., **Register** and **Un-Register**). **Note:** The button appears only in certain tables (e.g., Accounts table).
3	-	Added table rows displaying only some of the table parameters (columns).
4	-	Detailed view of a selected row, displaying all parameters.
5	-	Link to open the "child" table of the "parent" table. A link appears only if the table has a "child" table. The "child" table is opened for the selected row.
6	-	Navigation bar for scrolling through the table's pages (see Viewing Table Rows).
7	-	Search tool for searching parameters and values (see Searching Table Entries).
8	Edit	Modifies the selected row (see Modifying Table Rows).

Adding Table Rows

The following procedure describes how to add table rows. Before adding rows, the following GUI conventions are used:

- Commonly required parameters are displayed in **bold** font.
- If you change the value of a parameter (before clicking **Apply**), the parameter's field is highlighted, as shown in the example below:

- For indications of invalid values, see Invalid Value Indications.

➢ **To add a row:**

1. Click the **New** 

button, located on the table's toolbar; a dialog box appears.
2. Configure the parameters of the row as desired. For information on configuring parameters that are assigned a value which is a row referenced from another table, see Assigning Rows from Other Tables.
3. Click **Apply** to add the row to the table or click **Cancel** to ignore your configuration.
4. If the **Save** 

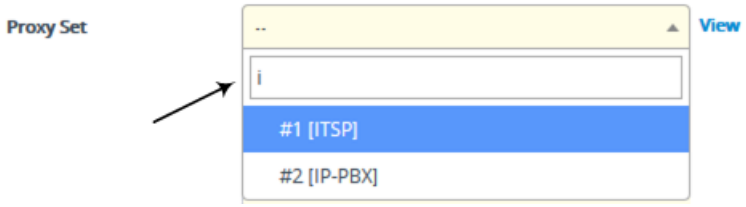
button is surrounded by a red border, you must save your settings to flash memory, otherwise they are discarded if the device resets (without a save to flash) or powers off.

Assigning Rows from Other Tables

Some tables contain parameters whose value is an assigned row (referenced-row) from another table (referenced-table). For example, the IP Groups table contains the 'Proxy Set' parameter whose value is an assigned Proxy Set, configured in the Proxy Sets table. These parameter types provide a drop-down list for selecting the value and a **View** button, as shown in the example below:

You can assign a referenced-row using one of the following methods:

- **Selecting a referenced-row from the drop-down list:**
  - Scroll down to the desired item and click it.
  - Search for the item by entering in the field the first few characters of the desired row, and then clicking it. The figure below shows an example of searched results for items (Proxy Sets) that begin with the letter "i":

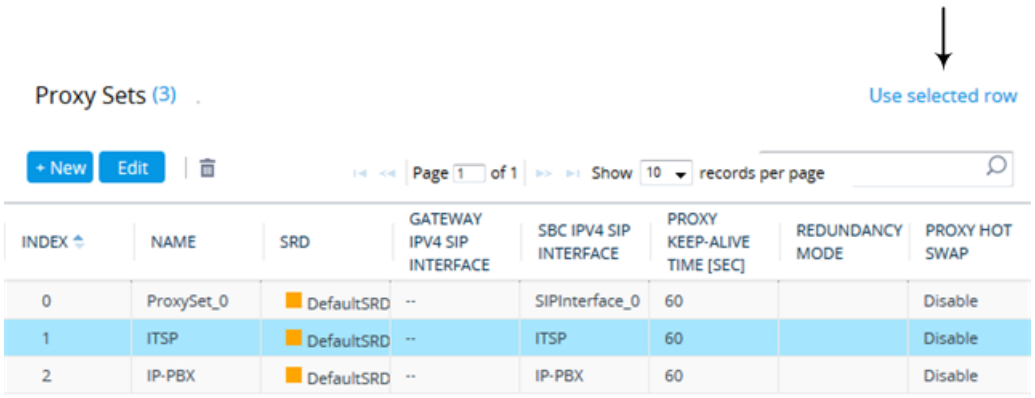


- **Selecting an existing referenced-row directly from the referenced-table:**

  a. Click **View**; the table (e.g., IP Groups table) and dialog box in which the button was clicked is minimized to the bottom-left corner of the Web interface and the referenced-table (e.g., Proxy Sets table) opens.

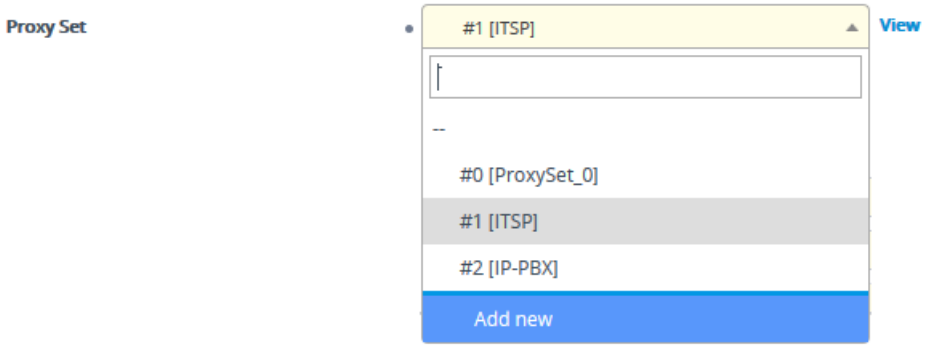
  b. Add a new row, if required; otherwise, skip this step.

  c. Select the desired row in the row-referenced table, and then click **Use selected row** located on the top-right of the table, as shown in the example below:



- **Adding a new referenced-row:**

  a. From the drop-down list, select the **Add new** option; as shown in the example below:



  The table (e.g., IP Groups table) and dialog box in which the **Add new** option was selected is minimized to the bottom-left corner of the Web interface and a dialog box appears for adding a new row in the referenced-table (e.g., Proxy Sets table).

  b. Configure the referenced-row and click **Apply**; the referenced-table (e.g., Proxy Sets table) closes and you are returned to the dialog box in which you selected the **Add new** option (e.g., IP Groups table), where the newly added row now appears selected.

You may want to access the referenced-table (e.g., Proxy Sets table) to simply view all its configured rows and their settings, without selecting one. To do this, click the **View** button. To return to the dialog box of the table (e.g., IP Groups table) in which you are making your configuration, click the arrow ↗ icon on the minimized dialog box to restore it to its previous size.

Modifying Table Rows

The following procedure describes how to modify (edit) the configuration of an existing table row. Remember that a gray-colored dot • icon displayed next to a parameter's value (as shown in the example below), indicates that it was changed from its default value:

➢ To edit a table row:

1. Select the row that you want to edit.
2. Click the **Edit** 

button, located on the table's toolbar; a dialog appears displaying the current configuration settings of the row.
3. Make your changes as desired, and then click **Apply**; the dialog box closes and your new settings are applied.
4. If the **Save** 

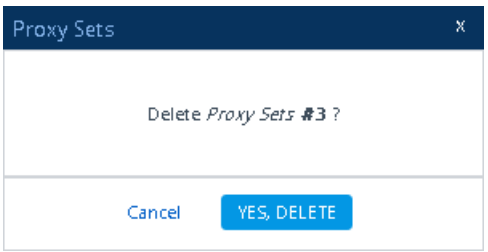
button is surrounded by a red border, you must save your settings to flash memory, otherwise they are discarded if the device resets (without a save to flash) or powers off.

Deleting Table Rows

The following procedure describes how to delete a row from a table.

➢ To delete a table row:

1. Select the row that you want to delete.
2. Click the delete 🗑 icon, located on the table's toolbar; a confirmation message box appears requesting you to confirm deletion, as shown in the example below:



3. Click **Yes, Delete**; the row is removed from the table and the total number of configured rows that is displayed next to the page title and page item in the Navigation tree is updated to reflect the deletion.

If the deleted row (e.g., a Proxy Set) was referenced in another table (e.g., IP Group), the reference is removed and replaced with an empty field. In addition, if the reference in the other table is for a mandatory parameter, the invalid ● icon is displayed where relevant. For example, if you delete a SIP Interface that you have assigned to a Proxy Set, the invalid icon appears alongside the **Proxy Sets** item in the Navigation tree as well as on the Proxy Sets page.

Invalid Value Indications

The Web interface provides the following indications of invalid values when configuring table rows:

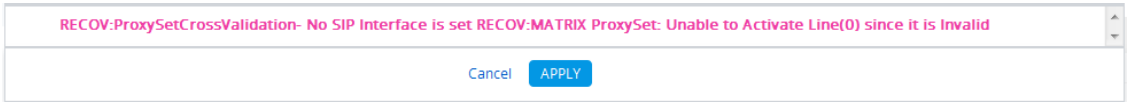
- **Parameters configured with invalid values:** An invalid value is a value that is not permissible for the parameter. This can include incorrect syntax (string, numeral, or character) or an out-of-range value. If you enter an invalid value and then click **Apply**, the field is surrounded by a colored border, as shown in the example below.

  If you hover your mouse over the field, a pop-up message appears providing the valid values. If you enter a valid value, the colored border is removed from the field. If you leave the parameter at the invalid value and click **Apply**, the parameter reverts to its previous value.

- **Mandatory parameters that reference rows of other configuration tables:**

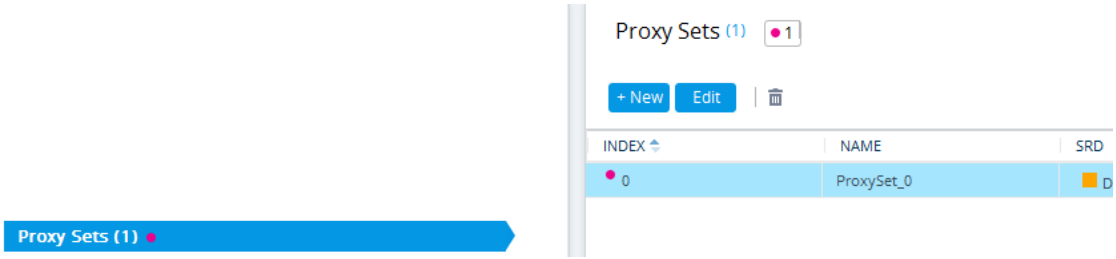
  - **Adding a row:** If you do not configure the parameter and you click **Apply**, an error message is displayed at the bottom of the dialog box. If you click **Cancel**, the dialog box closes and the row is not added to the table. For example, if you do not configure the 'SIP Interface' field (mandatory) for a Proxy Set (in the Proxy Sets table), the below message appears:

  - **Editing a row:** If you modify the parameter so that it's no longer referencing a row of another table (i.e., blank value), when you close the dialog box, the **Invalid Line** ● icon appears in the following locations:

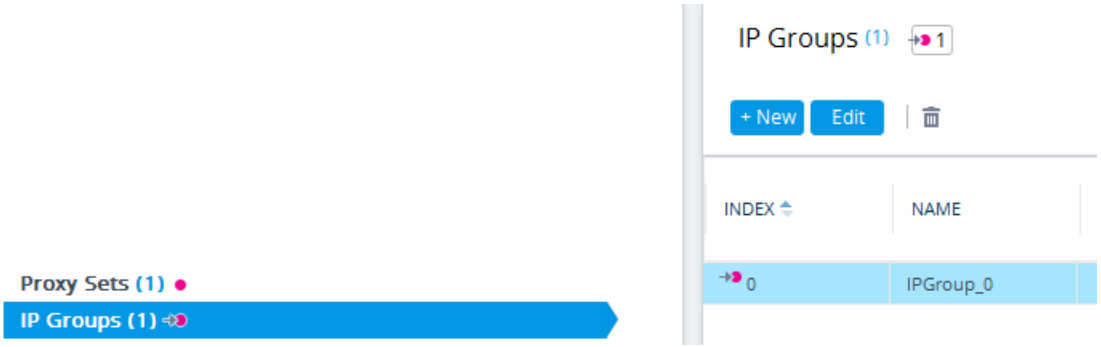
    - 'Index' column of the row.
    - Page title of the table. The total number of invalid rows in the table is also displayed with the icon.
    - Item in the Navigation tree that opens the table.

    For example, if you do not configure the 'SIP Interface' field (mandatory) for Proxy Set #0, the **Invalid Line** ● icon is displayed for the Proxy Sets table, as shown below:



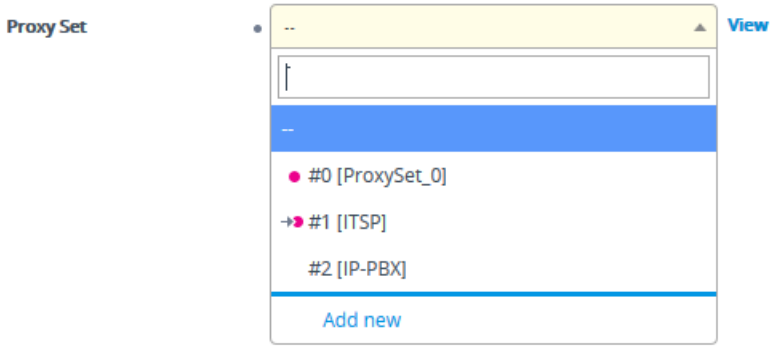
- **Parameters that reference rows of other configuration tables that are configured with invalid values:** If a row has a parameter that references a row of another table that has a parameter with an invalid value, the **Invalid Reference Line** icon is displayed in the following locations:
  - 'Index' column of the row.
  - Page title of the table. The total number of invalid rows in the table is also displayed with the icon.
  - Item in the Navigation tree that opens the table.

  For example, if you configure IP Group #0 (in the IP Groups table) with a parameter that references Proxy Set #0, which is configured with an invalid value, **Invalid Reference Line** icons are displayed for the IP Groups table, as shown below:



- **Invalid icon display in drop-down list items of parameters that reference rows of other tables:**
  - If the row has an invalid line (see description above), the **Invalid Line** icon appears along side the item.
  - If the row has an invalid reference line (see description above), the **Invalid Reference Line** icon appears along side it.

  For example, when configuring an IP Group, the 'Proxy Set' parameter's drop-down list displays items: Proxy Set #0 with indicating that it has an invalid parameter value, and Proxy Set #1 with indicating that it has a parameter that is referenced to a row of another table that has an invalid value:



If you assign a non-mandatory parameter with a referenced row and then later delete the referenced row (in the table in which the row is configured), the parameter's value automatically changes to an empty field (i.e., no row assigned). Therefore, make sure that you are aware of this and if necessary, assign a different referenced row to the parameter. Only if the parameter is mandatory is the **Invalid Line** icon displayed for the table in which the parameter is configured.

Viewing Table Rows

Tables display a certain number of rows per page. If you have configured more than this number, you can use the table's navigation bar to scroll through the table pages, as shown below and described in the subsequent table:

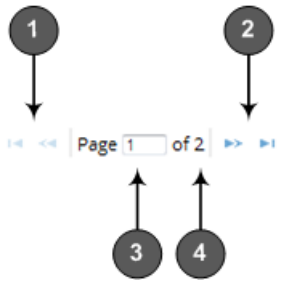


Table 6-3: Table Navigation Bar Description

Item	Description
1	Navigation buttons to view previous table rows: ■ Displays the previous table page ■ Displays the first table page (i.e., page with at least the first index row)
2	Navigation buttons to view the next table rows: ■ Displays the next table page ■ Displays the last table page (i.e., page with last index row)
3	Currently displayed table page. To open a specific table page, enter the page number and then press the Enter key.

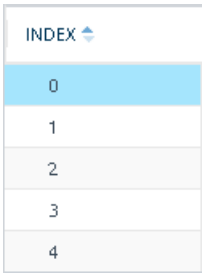
Item	Description
4	Total number of table pages.

Sorting Tables by Column

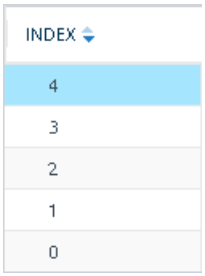
You can sort table rows by any column and in ascending order (e.g., 1, 2 and 3 / a, b, and c) or descending order (e.g., 3, 2, and 1 / c, b, and a). By default, most tables are sorted by the Index column and in ascending order.

➢ **To sort table rows by column:**

1. Click the name of the column by which you want to sort the table rows; the up-down arrows appear alongside the column name and the up button is displayed in a darker shade of color, indicating that the column is sorted in ascending order:



2. To sort the column in descending order, click the column name again; only the down arrow is displayed in a darker shade of color, indicating that the column is sorted in descending order:



Changing Index Position of Table Rows

You can change the position (index) of rows in tables. This is done by using the up-down arrows located on the table's toolbar.

- Changing row position can only done when the table is sorted by the 'Index' column and in ascending order; otherwise, the buttons are grayed out. For sorting table columns, see Sorting Tables by Column.
- Changing row position is supported only by certain tables (e.g., IP-to-IP Routing table).

➢ **To change the position of a row:**

1. Click the 'Index' column header so that the rows are sorted in ascending order (e.g., 0. 1, 2, and so on).
2. Select the row that you want to move.
3. Do one of the following:
   - To move one index up (e.g., from Index 3 to 2): Click the up ⬆ arrow; the row moves one index up in the table (e.g., to 2) and the row that originally occupied the index is moved one index down (e.g., to 3). In other words, the rows have swapped positions.
   - To move one index down (e.g., from Index 3 to Index 4): Click the down ⬇ arrow; the row moves one index down in the table (e.g., to 4) and the row that originally occupied the index is moved one index up (e.g., to 3). In other words, the rows have swapped positions.
4. Continue clicking the required arrow until the row has moved to the desired location in the table.

Searching Table Entries

You can search for any parameter value (alphanumeric) in configuration tables, using the Search tool. The Search tool, located above each table, is shown below and described in the subsequent table:

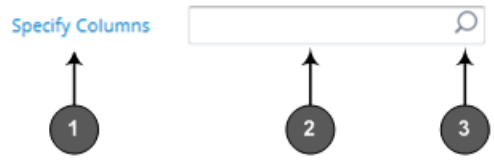


Table 6-4: Table Search Tool Description

Item#	Description
1	'Specify Columns' drop-down list for selecting the table column (parameter) in which to do the search. By default, the search is done in all columns.
2	Search box to enter your search key (parameter value).
3	Magnifying-glass icon which when clicked performs the search.

➢ **To search for a table value:**

1. If you want to perform the search on all table columns, skip this step; otherwise, from the 'Specify Columns' drop-down list, select the table column in which you want to perform the search; the name of the drop-down list changes to the name of the selected column.
2. In the Search box, enter the value for which you want to search.

3. Click the magnifying-glass icon to run the search. If the device finds the value, the table displays only the rows in which the value was found. You can then select any row and modify it by clicking the **Edit** button. If the search is unsuccessful, no rows are displayed.
4. To quit the Search tool and continue configuring rows, click the x icon located in the Search box.

Searching for Configuration Parameters

You can search in the Web interface for parameter names (standalone or table parameters) and values. The search key can include the full parameter name (Web or ini file name) or a substring of it. If you search for a substring, all parameters containing the substring in their names are listed in the search result. For example, to search for the parameter 'Telnet Server TCP Port', you can use any of the following search keys:

- "Telnet Server TCP Port" (Web name)
- "TelnetServerPort" (ini file name)
- "Telnet"
- "Port"

The search key for a parameter value can include alphanumerics and certain characters (see note below). The key can be a complete value or a partial value. The following are examples of search keys for searching values:

- "10.102.1.50"
- "10.15."
- "abc.com"
- "ITSP ABC"

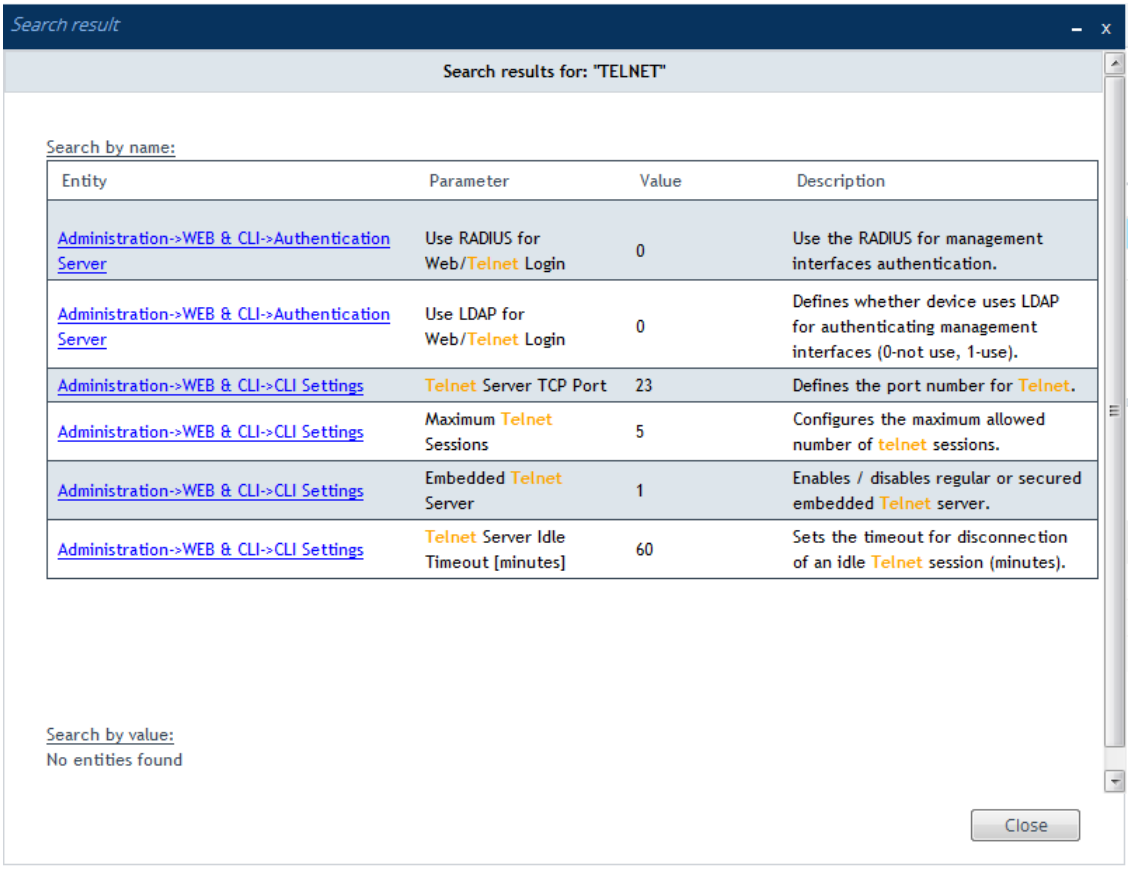
When the device completes the search, it displays a list of found results based on the search key. Each possible result, when clicked, opens the page on which the parameter or value is located. You need to click the most appropriate result.

> The search key can include only alphanumerics, periods, and spaces. The use of other characters are invalid.

➢ **To search for a parameter:**

1. In the search box, enter the search key (parameter name or value).
2. Click the search icon; the Search Result window appears, listing found parameters based on your search key. Each searched result displays the following:
   - Navigation path (link) to the page on which the parameter appears
   - Parameter's name
   - Parameter's value

- Brief description of parameter



3. Click the link of the navigation path corresponding to the required found parameter to open the page on which the parameter appears.

Getting Help

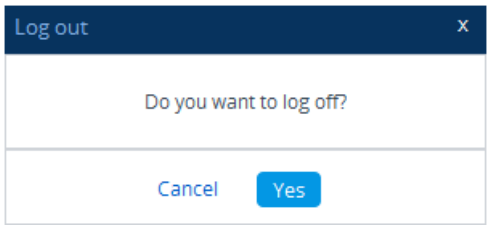
The Web interface provides you with context-sensitive pop-up help of standalone parameters. When you hover your mouse over a parameter's field, a pop-up appears with a short description of the parameter, as shown in the following example:

Logging Off the Web Interface

The following procedure describes how to log off the Web interface.

➢ **To log off the Web interface:**

1. On the menu bar, from the 'Admin' drop-down list, click **Log Out**; the following confirmation message box appears:



2. Click **Yes**; you are logged off the Web session and the Web Login window appears enabling you to re-login, if required.

Customizing the Web Interface

You can customize the following elements of the device's Web interface (GUI):

- Corporate logo (see Replacing the Corporate Logo)
- Device's (product) name (see Customizing the Product Name)
- Web browser tab label (see Customizing the Browser Tab Label on page 41)
- Web browser Favicon (see Customizing the Favicon)
- Login welcome message (see Creating a Login Welcome Message)

- The product name also affects other management interfaces.
- In addition to Web-interface customization, you can customize the following to reference your company instead of AudioCodes:
  - ✓ SNMP Interface: Product system OID (see the SNMPSysOid parameter) and trap Enterprise OID (see the SNMPTrapEnterpriseOid parameter).
  - ✓ SIP Messages: User-Agent header (see the UserAgentDisplayInfo parameter), SDP "o" line (see the SIPSDPSessionOwner parameter), and Subject header (see the SIPSubject parameter).

Replacing the Corporate Logo

You can replace the default corporate logo image (i.e., AudioCodes logo) that is displayed in the Web interface. The logo appears in the following Web areas:

- **Web Login screen:**

- **Menu bar:**

You can replace the logo with one of the following:

- A different image (see Replacing the Corporate Logo with an Image)
- Text (see Replacing the Corporate Logo with Text)

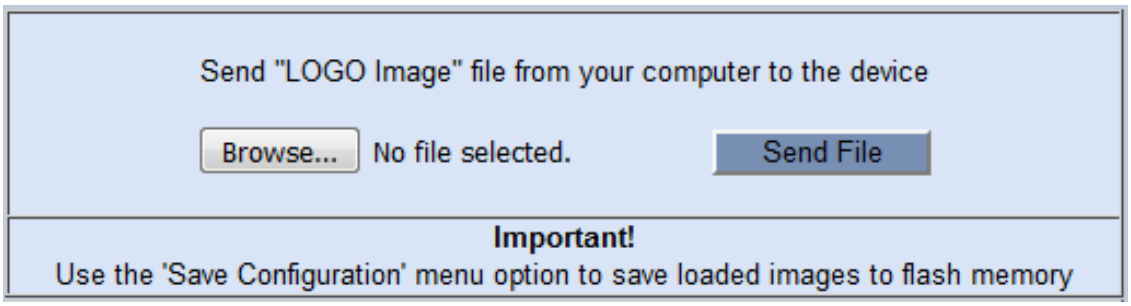
Replacing the Corporate Logo with an Image

You can replace the default corporate logo with a different image.

- The logo image file type can be GIF, PNG, JPG, or JPEG.
- The logo image must have a fixed height of 24 pixels. The width can be up to 199 pixels (default is 145).
- The maximum size of the image file can be 64 Kbytes.

➢ To replace the logo:

1. Save your new logo image file in a folder on the same PC that you are using to access the device's Web interface.
2. In your browser's URL address field, append the case-sensitive suffix "/AdminPage" to the device's IP address (e.g., http://10.1.229.17/AdminPage).
3. Log in with your credentials; the Admin page appears.
4. On the left pane, click **Image Load to Device**; the right pane displays the following:



5. Use the **Browse** button to select your logo file, and then click **Send File**; the device loads the file.
6. On the left pane, click **Back to Main** to exit the Admin page.
7. Reset the device with a save-to-flash for your settings to take effect.

Replacing Corporate Logo with Text

You can replace the logo with text. The following figure displays an example where the logo is replaced with the text, "My Text":

➢ **To replace logo with text:**

1. Create an ini file that includes the following parameter settings:

```
UseWebLogo = 1
WebLogoText = <your text, for example, My Text>
```

2. Load the ini file using the Auxiliary Files page (see Loading Auxiliary Files).
3. Reset the device with a save-to-flash for your settings to take effect.

> Make sure that the [LogoFileName] parameter is not configured to any value. If [LogoFileName] is configured, it overrides [UseWebLogo] and an image will always be displayed.

Replacing Text with Corporate Logo

If you have replaced the logo with text (as described in Replacing Corporate Logo with Text on the previous page), you can return the logo as described below.

➢ **To replace text with logo:**

1. Create an ini file that includes the following parameter settings:

```
UseWebLogo = 0
```

2. Load the ini file using the Auxiliary Files page (see Loading Auxiliary Files).
3. Reset the device with a save-to-flash for your settings to take effect.

Customizing the Browser Tab Label

You can customize the label that appears on the tab of the Web browser that you use to open the device's Web interface. By default, the tab displays "AudioCodes". You can change this to display either the IP address of the device or any customized text.

- You can customize the tab to display the device's IP address. This is applicable only if a logo image is used in the Web interface (see Replacing the Corporate Logo with an Image on page 40).
- If you are using the default AudioCodes corporate logo image in the Web interface, you can only customize the tab to display "AudioCodes" or the IP address.
- You can customize the tab to display text other than "AudioCodes", only if you are using a non-AudioCodes logo image in the Web interface.
- If you have replaced the corporate logo image with text (see Replacing Corporate Logo with Text on page 40), the same text is used for the tab.

➢ **To replace the browser tab label with the device's IP address:**

1. Create an ini file that includes the following parameter settings:

```
UseWebLogo = 1
WebLogoText =
```

If you have never configured the [WebLogoText] parameter, you can omit it from the ini file. If you have configured it before, then set it to an empty value, as shown above.

2. Load the ini file using the Auxiliary Files page (see Loading Auxiliary Files).
3. Reset the device with a save-to-flash for your settings to take effect.

➢ **To customize the text of the browser tab label:**

1. Create an ini file that includes the following parameter settings:
   - To replace the default text:

```
UseWebLogo = 1
WebLogoText = <your text, for example, Hello>
```

   - To restore the default text ("AudioCodes"):

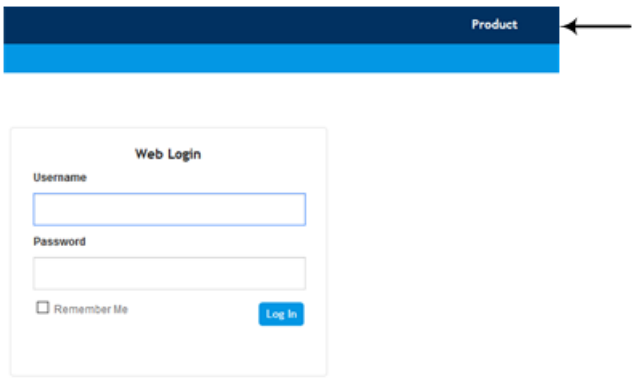
```
UseWebLogo = 0
```

2. Load the ini file using the Auxiliary Files page (see Loading Auxiliary Files).
3. Reset the device with a save-to-flash for your settings to take effect.

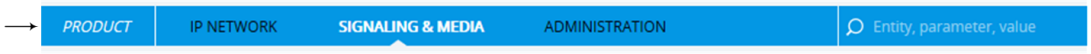
Customizing the Product Name

You can customize the device's product name. The name is displayed in various places in the management interfaces, as shown below using the example of the customized product name "Product":

- **Web Login screen:**



- **Web tab bar:**



- **ini file "Board" field:**

```
;Board: Product
```

- **CLI prompt:**

```
Product(config-system)#
```

➢ **To customize the device's product name:**

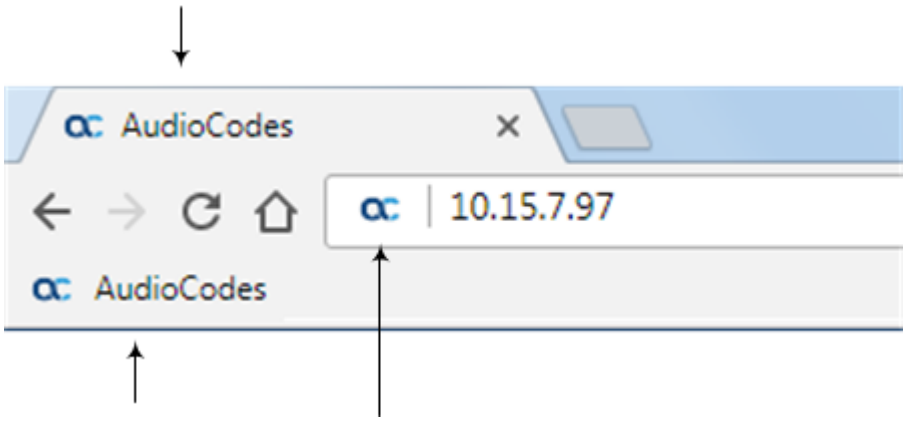
1. Create an ini file (*.ini) that includes the following parameter settings:

```
UseProductName = 1
UserProductName = < name >
```

2. Load the ini file using the Auxiliary Files page (see Loading Auxiliary Files).
3. Click the **Save** button on the toolbar to save your settings to flash memory.

Customizing the Favicon

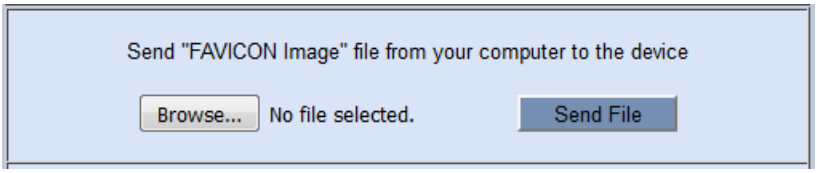
You can replace the default favicon with your own personalized favicon. Depending on the browser, the favicon is displayed in various areas of your browser, for example, in the URL address bar, on the page tab, and when bookmarked.



- The logo image file type can be ICO, GIF, or PNG.
- The maximum size of the image file can be 16 Kbytes.

➢ **To customize the favicon:**

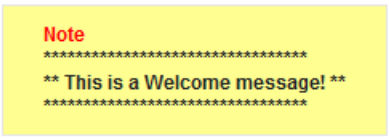
1. Save your new favicon file in a folder on the same PC that you are using to access the device's Web interface.
2. In your browser's URL address field, append the case-sensitive suffix "/AdminPage" to the device's IP address (e.g., http://10.1.229.17/AdminPage).
3. Log in with your credentials; the Admin page appears.
4. On the left pane, click **Image Load to Device**; the right pane displays the following:

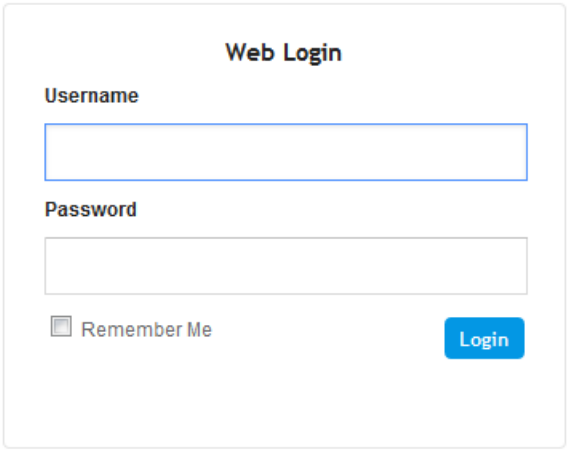


5. Use the **Browse** button to select your favicon file, and then click **Send File**; the device loads the image file.
6. On the left pane, click **Back to Main** to exit the Admin page.
7. Reset the device with a save-to-flash for your settings to take effect.

Creating a Login Welcome Message

You can create a personalized welcome message that is displayed on the Web Login screen. The message always begins with the title "Note" and has a color background, as shown in the example below:





➢ To create a login welcome message:

1. Using a text-based editor (e.g., Notepad) to create an ini file that includes only the [WelcomeMessage] table parameter. Use the parameter to configure your message, where each index row is a line in your message, for example:

```
[WelcomeMessage ]
FORMAT WelcomeMessage_Index = WelcomeMessage_Text;
WelcomeMessage 1 = "**********";
WelcomeMessage 2 = "** This is a Welcome message! **";
WelcomeMessage 3 = "**********";
[\WelcomeMessage]
```

2. Upload the ini file to the device through the Auxiliary Files page (see Loading Auxiliary Files).
3. Save your new configuration to flash.

> Uploading an ini file through the Auxiliary Files page doesn't require a device reset.

➢ To remove the welcome message:

1. Download the device's configuration as an ini file through the Configuration File page (see Downloading and Loading ini Configuration File on page 895).
2. Open the file in a text-based editor, remove the [WelcomeMessage] table, and then save the file.

3. Upload the file through the Configuration File page.

> After the file is uploaded, the device resets to apply your new configuration.

Configuring Additional Management Interfaces

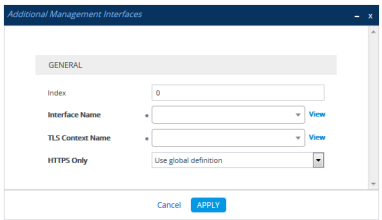
The Additional Management Interfaces table lets you configure up to 16 management interfaces, in addition to the **OAMP** management interface configured in the IP Interfaces table. Multiple management interfaces lets you remotely access the device's management interfaces (see note below) through different IP addresses. Each additional management interface can be configured to use a specific network interface (Control or Media type, or both) and TLS Context, and can be configured to restrict access through HTTPS only.

- The feature is applicable only to the device's Web- and REST-based management interfaces.
- To allow access to the device's management interfaces through all network interfaces configured in the IP Interfaces table, see the [EnableWebAccessFromAllInterfaces] parameter. This parameter does not specify a TLS Context nor a connectivity protocol (HTTP or HTTPS).
- For secured management of the default management network interface (i.e., **OAMP** Application Type in the IP Interfaces table), the device uses the default TLS Context (Index #0 and named "default").

The following procedure describes how to configure additional management interfaces through the Web interface. You can also configure it through ini file [AdditionalManagementInterfaces] or CLI (`configure system > additional-mgmt-if`).

➢ **To configure additional management interfaces:**

1. Open the Additional Management Interfaces table (**Setup** menu > **Administration** tab > **Web & CLI** folder > **Additional Management Interfaces**).
2. Click **New**; the following dialog box is displayed:



3. Configure an additional management interface according to the parameters described in the table below.
4. Click **Apply**, and then save your settings to flash memory.

Table 6-5: Additional Management Interfaces Table Parameter Descriptions

Parameter	Description
General	
'Index' [AdditionalManagementInterfaces_Index]	Defines an index number for the new table row. **Note:** Each row must be configured with a unique index.
'Interface Name' `interface-name` [AdditionalManagementInterfaces_InterfaceName]	Assigns an IP network interface (from the IP Interfaces table) to the management interface. For more information on IP network interfaces, see Configuring IP Network Interfaces. **Note:** ■ Only Control- and/or Media-type IP network interfaces can be associated with additional management interfaces. ■ An IP network interface can be associated with only **one** additional management interface.
'TLS Context Name' `tls-context-name` [AdditionalManagementInterfaces_TLSContextName]	Assigns a TLS Context (from the TLS Contexts table) to the management interface. A TLS Context provides secure TLS-based management access. For more information on TLS Contexts, see Configuring TLS Certificate Contexts.
'HTTPS Only' `https-only-val` [AdditionalManagementInterfaces_HTTPSOnly]	Defines the protocol required for accessing the management interface. ■ [0] **HTTP and HTTPS** = The management interface can be accessed over a secured (HTTPS) and an unsecured (HTTP) connection. ■ [1] **HTTPS Only** = The management interface can be accessed only over a secured (HTTPS) connection. ■ [2] **Use global definition** = The type of management connection (HTTP and HTTPS, or HTTPS Only) depends on the configuration of the global parameter [HTTPSOnly], as described in Configuring Secured (HTTPS) Web.

Configuring Management User Accounts

The Local Users table lets you configure up to 20 management user accounts for the device's management interfaces (Web interface, CLI and REST API).

You configure each user account with login credentials (username and password) and a management user level which defines the level of read and write privileges. The table below describes the different types of user levels.

Table 6-6: Description of Management User Levels

User Level	Numeric Representation in RADIUS	Privileges
Security Administrator	200	■ Read-write to all Web pages. ■ Read-write (access) to the CLI's Privileged User mode (`> enable`). ■ Create all other user levels. **Note:** ■ At least one Security Administrator user must exist. ■ Only the Security Administrator can create the **first** Master user. Once created, additional Master users can only be created or deleted by other Master users.
Master	220	■ Read-write to all Web pages. ■ Read-write (access) to the CLI's Privileged User mode (`> enable`). ■ Create all user levels (including Security Administrators). ■ Delete all users except the last Security Administrator. ■ Create or delete Master users. **Note:** ■ Only the Security Administrator can create the **first** Master user. Once created, additional Master users can only be created or deleted by other Master users. ■ If only one Master user exists, it can be

User Level	Numeric Representation in RADIUS	Privileges
		deleted only by itself.
Administrator	100	Read-write to all Web pages, except security-related pages (including the Local Users table) where this user has read-only privileges. **Note:** This user level can access only the CLI's Basic User mode.
Monitor	50	Read-only, but access to security-related pages (including the Local Users table) is blocked. **Note:** This user level can access only the CLI's Basic User mode.

- Only Security Administrator and Master users can configure users in the Local Users table.
- For privileges per user level for the device's REST API, refer to the document, *REST API for SBC-Gateway-MSBR Devices*.
- Regardless of user level, all users can change their login password as described in Changing Login Password by All User Levels on page 63.
- You can change the read-write and read-only privileges per Web page for Monitor, Administrator, and Security Administrator user levels. For more information, see Customizing Access Levels per Web Page on page 56.

The device provides the following two default user accounts:

Table 6-7: Default User Accounts

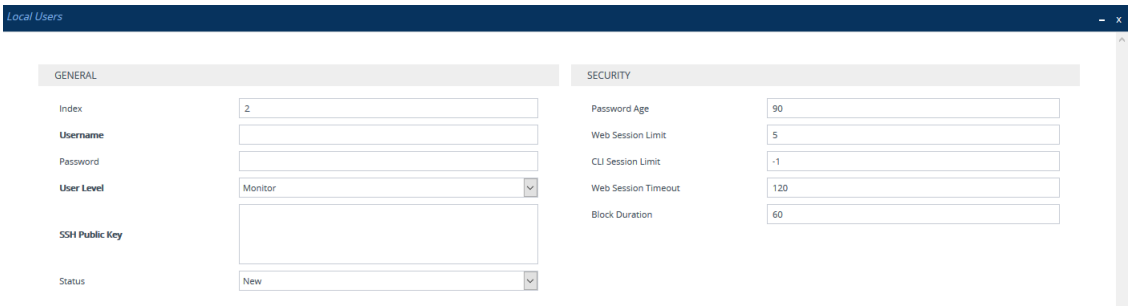
User Level	Username (Case-Sensitive)	Password (Case-Sensitive)
Security Administrator	"Admin"	"Admin"
Monitor	"User"	"User"

- For security, it's recommended that you change the default username and password of the default users.
- To restore the device to these default users (and with their default usernames and passwords), configure the [ResetWebPassword] parameter to [1]. All other configured accounts are deleted.
- If you want to use the same Local Users table configuration for another device, before uploading this device's configuration file (.ini) to the other device, you **must** edit the file so that the passwords are in plain text.
- If you delete a user who is currently in an active Web session, the user is immediately logged off the device.
- Up to five users can be concurrently logged in to the Web interface; they can all be the same user.
- You can set the entire Web interface to read-only (regardless of Web user access levels), using the [DisableWebConfig] parameter (see Web and Telnet Parameters).
- You can configure additional Web user accounts using a RADIUS server (see RADIUS Authentication).

The following procedure describes how to configure user accounts through the Web interface. You can also configure it through ini file [WebUsers] or CLI (`configure system > user`).

➢ **To configure management user accounts:**

1. Open the Local Users table (**Setup** menu > **Administration** tab > **Web & CLI** folder > **Local Users**).
2. Click **New**; the following dialog box is displayed:



3. Configure a user account according to the parameters described in the table below.
4. Click **Apply**, and then save your settings to flash memory.

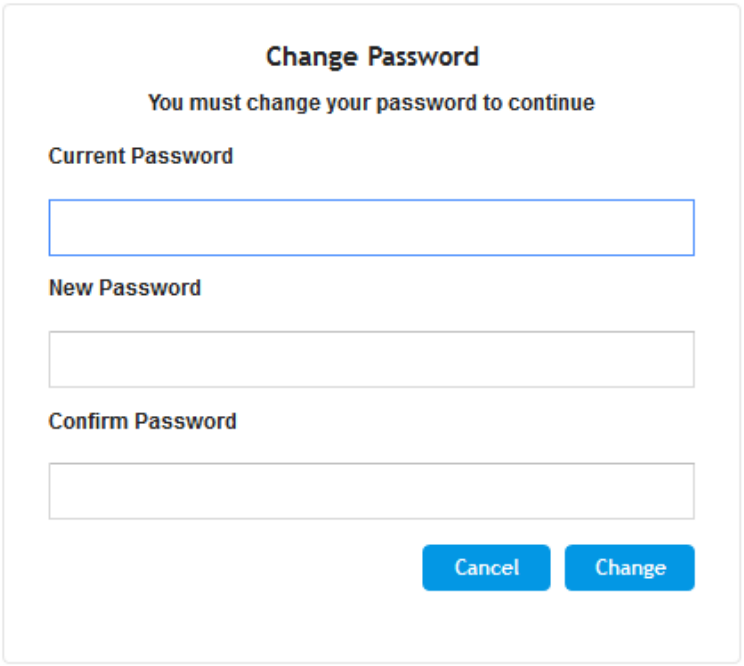
Table 6-8: Local Users Table Parameter Descriptions

Parameter	Description
General	
'Index' [WebUsers_Index]	Defines an index number for the new table row. **Note:** Each row must be configured with a unique index.

Parameter	Description
'Username' `user` [WebUsers_Username]	Defines the Web user's username. The valid value is a string of up to 40 alphanumeric characters, including the period ".", underscore "_", and hyphen "-" signs.
'Password' `password` [WebUsers_Password]	Defines the Web user's password. The valid value is a string of 8 to 40 ASCII characters. To ensure strong passwords, adhere to the following password complexity requirements: ■ The password should contain at least eight characters. ■ The password should contain at least two uppercase letters (e.g., A). ■ The password should contain at least two lowercase letters (e.g., a). ■ The password should contain at least two numbers (e.g., 4). ■ The password should contain at least two symbols (non-alphanumeric characters, e.g., $, #, %). ■ The password must not contain any spaces. ■ The password should contain at least four new characters that were not used in the previous password. To **enforce** password complexity requirements as listed above, configure the [EnforcePasswordComplexity] to [1]. If you enable password complexity, you can also configure the minimum length (number of characters) of the password, using the [MinWebPasswordLen] parameter. **Note:** ■ The password must **not** contain a backslash (\). ■ For security, password characters are not shown in the Web interface or ini file. In the Web interface, they are displayed as dots when you enter the password and then once applied, the password is displayed as an asterisk (*) in the table. In the ini file, they are displayed as an encrypted string. ■ To enforce obscured (encrypted) passwords when configuring the Local Users table through CLI, see the [CliObscuredPassword] parameter. ■ The password cannot be configured with wide characters.

Parameter	Description
'User Level' `privilege` [WebUsers_UserLevel]	Defines the user's access level. ■ **Monitor** = (Default) Read-only user. This user can only view Web pages and access to security-related pages is denied. ■ **Administrator** = Read/write privileges for all pages except security-related pages including the Local Users table where this user has read-only privileges. ■ **Security Administrator** = Full read/write privileges for all pages. ■ **Master** = Read/write privileges for all pages. This user also functions as a security administrator. **Note:** ■ At least one Security Administrator must exist. You cannot delete the last remaining Security Administrator. ■ The first Master user can be added only by a Security Administrator user. ■ Additional Master users can be added, edited and deleted only by Master users. ■ If only one Master user exists, it can be deleted only by itself. ■ Master users can add, edit, and delete Security Administrators (except the last Security Administrator). ■ Only Security Administrator and Master users can add, edit, and delete Administrator and Monitor users.
'SSH Public Key' `public-key` [WebUsers_ SSHPublicKey]	Defines a Secure Socket Shell (SSH) public key for RSA public-key authentication (PKI) of the remote user when logging into the device's CLI through SSH. Connection to the CLI is established only when a successful handshake with the user's private key occurs. The valid value is a string of up to 512 characters. By default, no value is defined. **Note:** ■ For more information on SSH and for enabling SSH, see Enabling SSH with RSA Public Key for CLI. ■ To configure whether SSH public keys are optional or mandatory, use the SSHRequirePublicKey parameter.

Parameter	Description
'Status' status [WebUsers_Status]	Defines the status of the user. ■ **New** = (Default) User is required to change its password on the next login. When the user logs in to the Web interface, the user is immediately prompted to change the current password. ■ **Valid** = User can log in to the Web interface as normal. ■ **Failed Login** = The state is automatically set for users that exceed a user-defined number of failed login attempts, set by the 'Deny Access on Fail Count' parameter (see Configuring Web Session and Access Settings). These users can log in only after a user-defined timeout configured by the 'Block Duration' parameter (see below) or if their status is changed (to New or Valid) by a Security Administrator or Master. ■ **Inactivity** = The state is automatically set for users that have not accessed the Web interface for a user-defined number of days, set by the 'User Inactivity Timer' (see Configuring Web Session and Access Settings). These users can only log in to the Web interface if their status is changed (to New or Valid) by a System Administrator or Master. **Note:** ■ The **Inactivity** option is applicable only to Administrator and Monitor users; Security Administrator and Master users can be inactive indefinitely. ■ If there is only one Security Administrator user, you cannot configure it to **Inactivity**; at least one Security Administrator must be **Valid**. ■ For security, it is recommended to set the status of a newly added user to **New** in order to enforce password change.
Security	
'Password Age' password-age [WebUsers_ PwAgeInterval]	Defines the duration (in days) of the validity of the password. When the duration elapses (i.e., password expires), when attempting to log in, the user is prompted to change the password (shown below), and then log in with the new password; otherwise, access to the Web interface is blocked.

Parameter	Description
	 The valid value is 0 to 10000, where 0 means that the password is always valid. The default is 90. **Note:** After logging in with your new password, you must save your settings, by clicking the **Save** button on the Web interface's toolbar. If not, the next time you attempt to log in, you will be prompted again to change the expired password.
'Web Session Limit' `session-limit` [WebUsers_ SessionLimit]	Defines the maximum number of concurrent Web interface and REST sessions allowed for the specific user account from different management stations / computers (IP addresses) or different Web browsers. For example, if configured to 2, the user account can be logged into the device's Web interface (i.e., same username-password combination) from two different management stations (i.e., IP addresses), or from two different Web browsers (e.g., Google Chrome and Microsoft Edge) at the same time. Once the user logs into the device, the session is active until the user logs off or until the session expires if the user is inactive for a user-defined duration (see the 'Web Session Timeout' parameter below). The valid value is 0 to 10. The default is 5. A value of 0 means that no sessions are allowed (see note below regarding REST). **Note:** ■ If you configure the parameter, when you click **Apply**

Parameter	Description
	you're automatically logged out of the Web session (and can log in again if configured to any value other than 0). ■ Closing the Web browser's window (by clicking the window's **x** button) doesn't end the session. Therefore, whenever you finish using the Web interface, it's recommended to log out of the Web interface to end your session. ■ If the number of concurrently logged-in users is at maximum, the device allows an additional user to log in through REST.
'CLI Session Limit' `cli-session-limit` [WebUsers_ CliSessionLimit]	Defines the maximum number of concurrent CLI sessions allowed for the specific user. For example, if configured to 2, the same user account can be logged into the device's CLI (i.e., same username-password combination) from two different management stations (i.e., IP addresses) at any one time. Once the user logs in, the session is active until the user logs off or until the session expires if the user is inactive for a user-defined duration (see the 'Web Session Timeout' parameter below). The valid value is -1, or 0 to 100. The default is -1, which means that the limit is according to the global parameters, 'Maximum Telnet Sessions' (TelnetMaxSessions) or 'Maximum SSH Sessions' (SSHMaxSessions).
'Web Session Timeout' `session-timeout` [WebUsers_ SessionTimeout]	Defines the duration (in minutes) of inactivity of a logged-in user in the Web interface, after which the user is automatically logged off the Web session. In other words, the session expires when the user has not performed any operations (activities) in the Web interface for the configured timeout duration. The valid value is 0, or to 100000. A value of 0 means no timeout. The default value is according to the settings of the WebSessionTimeout global parameter (see Configuring Web Session and Access Settings).
'Block Duration' `block-duration` [WebUsers_BlockTime]	Defines the duration (in seconds) for which the user is blocked when the user exceeds the maximum number of allowed failed login attempts, configured by the global parameter, 'Deny Access On Fail Count' [DenyAccessOnFailCount] parameter (see Configuring Web Session and Access Settings). The valid value is 0 to 100000, where 0 means that the user can do as many login failures without getting blocked. The

Parameter	Description
	default is according to the settings of the global parameter, 'Deny Authentication Timer' [DenyAuthenticationTimer] parameter (see Configuring Web Session and Access Settings). **Note:** The 'Deny Authentication Timer' parameter relates only to failed Web logins from specific IP addresses (management stations), which configures the interval (in seconds) that the user needs to wait before logging into the device from the same IP address after reaching the maximum number of failed login attempts.

Customizing Access Levels per Web Page

The Customize Access Level table lets you configure up to 100 customized access rules. These rules assign read-write (view and configure) or read-only (view) privileges to management user levels (Monitor, Administrator, or Security Administrator) per page in the device's Web interface. These rules override the default read-write and read-only privileges of these user levels (as described in Configuring Management User Accounts on page 48). Whatever user level is specified, the rule applies to that level and all levels that are higher than that level (Security Administrator is the highest user level and Monitor is the lowest user level). If you attempt to open a page for which you don't have access privileges, the page displays the message "Your access level does not allow you to view this page".

For security reasons, some pages (e.g., the TLS Contexts page) cannot be customized in this table.

The following table provides a few configuration examples to facilitate your understanding of assigning read-write and read-only privileges to user levels per Web page.

Page Name	Read-Write Access Level	Read-Only Access Level	Description
CLI Settings	Monitor	Monitor	Assigns read-write (and read-only) privileges to Monitor users for the CLI Settings page. As this is the lowest user level, it means that all higher user levels (i.e.,

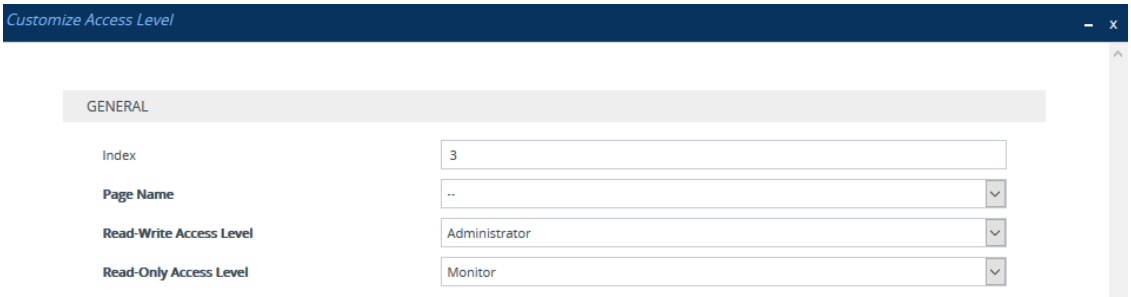
Page Name	Read-Write Access Level	Read-Only Access Level	Description
			Administrator and Security Administrator) are also assigned full read-write privileges.
Firewall	Security Administrator	Monitor	Assigns read-write privileges to Security Administrator users for the Firewall page. As this is the highest user level, only Security Administrator users have write privileges for this page. In addition, as this rule assigns read-only privileges to Monitor users, which is the lowest user level, all higher user levels (i.e., Administrator and Security Administrator) are also assigned read-only privileges.
TLS Contexts	Security Administrator	Security Administrator	Assigns read-write privileges to Security Administrator users for the TLS Contexts page.

Page Name	Read-Write Access Level	Read-Only Access Level	Description
			As this is the highest user level, no other user level can access (read) or configure (write) this page.

The following procedure describes how to configure customized access level rules through the Web interface. You can also configure it through ini file [WebPagesAccessLevel].

➢ **To customize access levels:**

1. Open the Customize Access Level table (**Setup** menu > **Administration** tab > **Web & CLI** folder > **Customize Access Level**).
2. Click **New**; the following dialog box is displayed:



3. Configure the rule according to the parameters described in the table below.
4. Click **Apply**, and then save your settings to flash memory.

Table 6-9: Customize Access Level Table Parameter Descriptions

Parameter	Description
'Index' [WebPagesAccessLevel_Index]	Defines an index number for the new table row. **Note:** Each row must be configured with a unique index.
'Page Name' [WebPagesAccessLevel_ PageNameFromTree]	Defines the Web page whose access level you want to customize. **Note:** For security reasons, some pages are not listed under this parameter and therefore, cannot be customized.
'Read-Write Access Level'	Defines the **minimum** user level to which you want to

Parameter	Description
[WebPagesAccessLevel_ RWAccessLevel]	assign read-write access privileges for the selected Web page. ■ [50] **Monitor** ■ [100] **Administrator** (default) ■ [200] **Security Administrator**
'Read-Only Access Level' [WebPagesAccessLevel_ ROAccessLevel]	Defines the **minimum** user level to which you want to assign read-only access privileges for the selected Web page. ■ [50] **Monitor** (default) ■ [100] **Administrator** ■ [200] **Security Administrator** **Note:** The user level must be the same or lower than the user level you configured in the 'Read-Write Access Level' parameter. For example, you cannot assign read-only privileges to the Security Administrator if you have assigned read-write privileges to the Administrator.

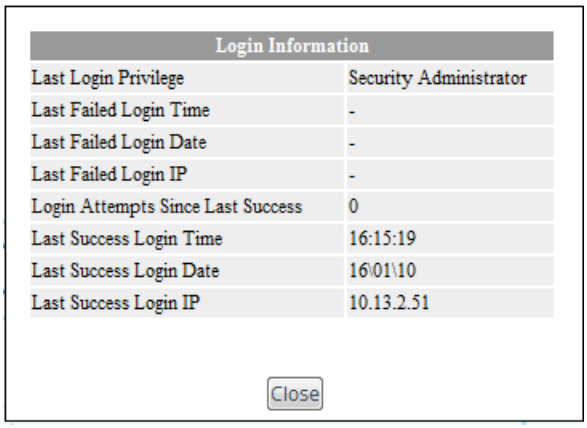
Displaying Login Information upon Login

You can enable the device to display login information immediately upon Web login.

➢ **To enable display of user login information upon login:**

1. Open the Web Settings page (**Setup** menu > **Administration** tab > **Web & CLI** folder > **Web Settings**).
2. Under the **Security** group, from the 'Display Last Login Information' drop-down list, select **Enable**.
3. Click **Apply**.

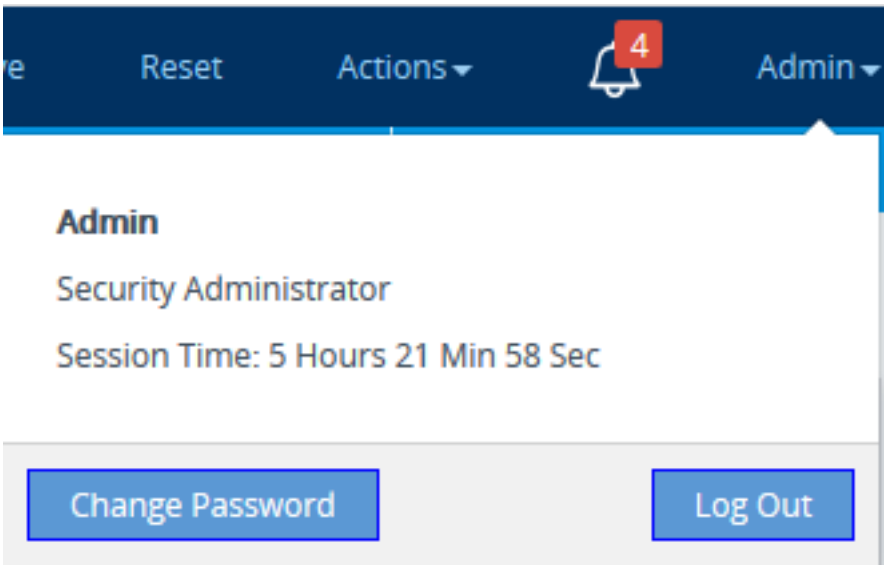
Once enabled, each time you login to the device, the Login Information window is displayed, as shown in the example below:



To close the window, click **Close**.

Viewing Logged-In User Information

The username of the currently logged in user is displayed in the top-right corner of the Web interface. If you click the username (e.g., "Admin"), a drop-down box appears, for example:



The following information is displayed:

- Username (e.g., Admin) of currently logged-in user
- User level (e.g., Security Administrator) of currently logged-in user
- Duration of the current Web session (starting from login)

The following buttons are also displayed:

- **Log Out**: Logs you out of the Web session (see Logging Off the Web Interface)
- **Change Password**: Allows you to change your login password (see Changing Login Password by All User Levels on page 63)

Configuring Web Session Timeouts

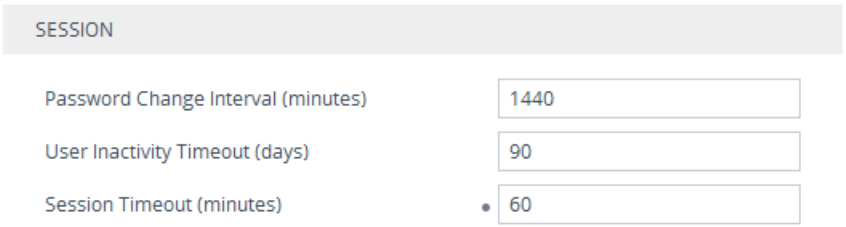
You can configure various user timeouts for the device's Web interface:

- Session timeout, where the user is automatically logged out of the Web interface if the user is inactive for a user-defined duration.
- Logged-in timeout, where the user is blocked from logging in if the user has not logged into the Web interface within a user-defined duration.

> **Note:** You can only perform the configuration described in this section if you are a management user with Security Administrator level or Master level. For more information, see Configuring Management User Accounts.

➢ **To configure Web user sessions and access security:**

1. Open the Web Settings page (**Setup** menu > **Administration** tab > **Web & CLI** folder > **Web Settings**).
2. Under the **Session** group, configure the following parameters:



   - 'User Inactivity Timeout': If the user has not logged into the Web interface within this duration, the status of the user becomes inactive and the user can no longer access the Web interface. The user can only log in to the Web interface if its status is changed (to **New** or **Valid**) by a Security Administrator or Master user (see Configuring Management User Accounts).
   - 'Session Timeout': Defines the duration (in minutes) of inactivity (i.e., no actions are performed in the Web interface) of a logged-in user, after which the Web session expires and the user is automatically logged off the Web interface and needs to log in again to continue the session. You can also configure the functionality per user in the Local Users table (see Configuring Management User Accounts), which overrides this global setting.
3. Click **Apply**.

For a detailed description of the above parameters, see Web Parameters.

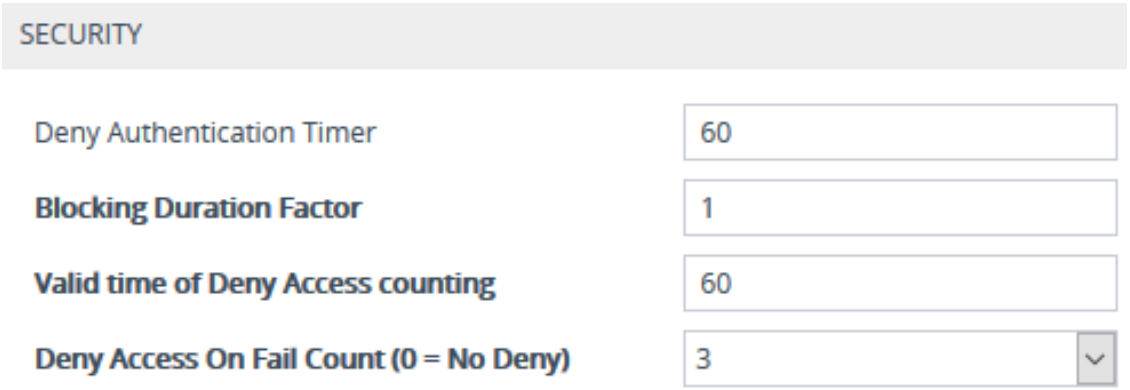
Configuring Deny Access for Failed Login Attempts

You can configure the device to block users or management stations (IP addresses) from accessing the web interface if the user enters incorrect login credentials for a user-defined number of successive login attempts.

> You can only perform the configuration described in this section if you are a management user with Security Administrator level or Master level. For more information, see Configuring Management User Accounts.

➢ **To configure deny access upon failed login attempts:**

1. Open the Web Settings page (**Setup** menu > **Administration** tab > **Web & CLI** folder > **Web Settings**).
2. Under the **Security** group, configure the following parameters:



- 'Deny Authentication Timer' [DenyAuthenticationTimer]: Define the duration (in seconds) for which login to the Web interface is denied from a specific IP address (management station) for **all** users, when the number of failed login attempts has exceeded the maximum. To configure the blocked duration **per user**, use the 'Block Duration' [WebUsers_BlockTime] parameter in the Local Users table (see Configuring Management User Accounts).
- 'Blocking Duration Factor' [BlockDurationFactor]: Define the number to multiple the previous blocking time for blocking the IP address or the user upon the next failed login scenario.
- 'Value time of Deny Access counting' [DenyAccessCountingValidTime]: Defines the maximum time interval (in seconds) between failed login attempts to be included in the count of failed login attempts for denying access to the user.
- 'Deny Access On Fail Count' [DenyAccessOnFailCount]: Define the maximum number of failed login attempts, after which the requesting IP address (management station) for all users is blocked.

For a detailed description of the parameters mentioned above, see Web Parameters on page 1129.

3. Click **Apply**.

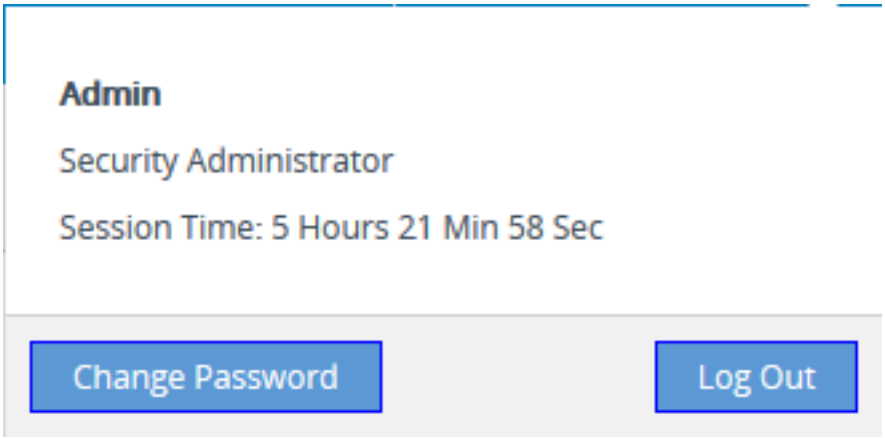
Changing Login Password by All User Levels

Regardless of your user level (e.g., Monitor or Administrator), you can change your login password through the Change Password dialog box, accessed from the Web interface's top bar.

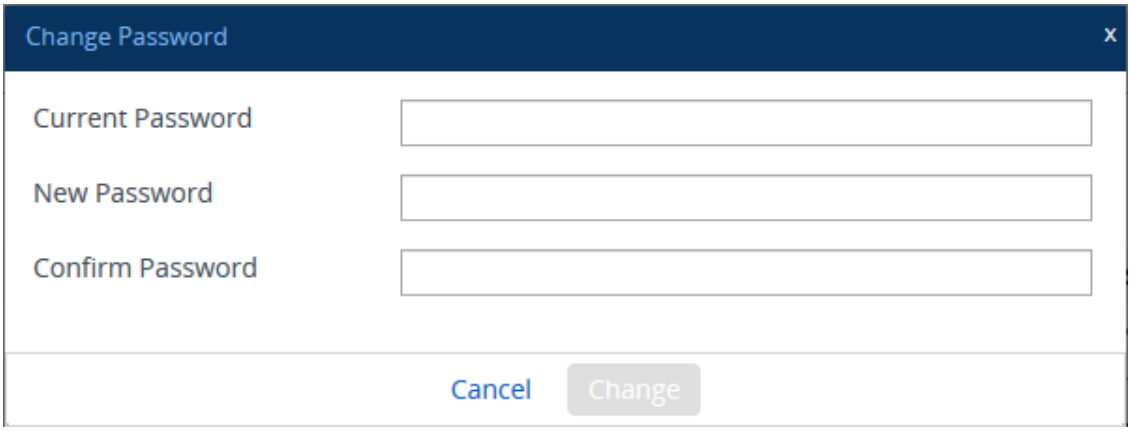
- Users with Security Administrator level or Master level can also change passwords for themselves and for other user levels in the Local Users table (see Configuring Management User Accounts).
- For valid passwords, see the 'Password' parameter in the Local Users table.
- You can only change the password if the duration, configured by the 'Password Change Interval' parameter (Web Settings page - **Setup** menu > **Administration** tab > **Web & CLI** folder > **Web Settings**), has elapsed since the last password change.

➢ **To change the login password:**

1. On the top bar of the Web interface, click the username that is displayed for the currently logged-in user (e.g., "Admin"); the following appears:



2. Click **Change Password**; the following appears:



3. In the 'Current Password' field, enter your current login password.
4. In the 'New Password' field, enter your new password.
5. In the 'Confirm Password' field, enter your new password again.
6. Click **Change**; you are logged off the Web session and prompted to log in again with your new password.

Configuring Secured (HTTPS) Web

By default, the device allows remote management (client) through HTTP and HTTPS. However, you can enforce secure Web access communication by configuring the device to accept only HTTPS requests.

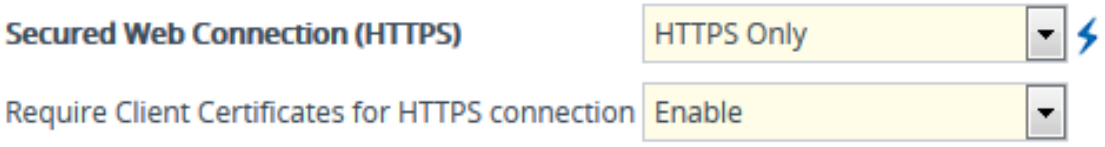
By default, servers using TLS provide one-way authentication. The client is certain that the identity of the server is authentic. However, when an organizational Public Key Infrastructure (PKI) is used, two-way authentication (TLS mutual authentication) may be desired; both client and server should be authenticated using X.509 certificates. This is achieved by installing a client certificate on the management PC and loading the Certification Authority's (CA) root certificate to the device's Trusted Certificates table (certificate root store). The Trusted Root Certificate file may contain more than one CA certificate combined, using a text editor.

- For secured management through the device's default management network interface (i.e., **OAMP** Application Type in the IP Interfaces table), the device uses the default TLS Context (Index #0 and named "default"). However, for secured Web- and REST-based management through Additional Management Interfaces (configured in Configuring Additional Management Interfaces on page 46), you can use any TLS Context.
- The 'Secured Web Connection (HTTPS)' parameter (mentioned below) is also applicable to REST-based management.

➢ **To configure secure (HTTPS) Web access:**

1. Open the Web Settings page (**Setup** menu > **Administration** tab > **Web & CLI** folder > **Web Settings**), and then do the following.
   - From the 'Secured Web Connection (HTTPS)' drop-down list, select **HTTPS Only**.

- To enable two-way authentication whereby both management client and server are authenticated using X.509 certificates, from the 'Require Client Certificates for HTTPS connection' drop-down list, select **Enable**.



2. If you want to configure secured management through an Additional Management Interface (i.e., not through the default management network interface called **OAMP** in the IP Interfaces table), then configure an Additional Management Interface as described in Configuring Additional Management Interfaces on page 46. Assign it a TLS Context and enable it for **HTTPS Only**.
3. **(TLS Mutual Authentication Only)** In the TLS Contexts table (see Configuring TLS Certificate Contexts), select the required TLS Context (see following note), and then click the **Trusted Root Certificates** link located below the table; the Trusted Certificates table appears.

> If you are securing management through the default management network interface (i.e., **OAMP** in the IP Interfaces table), then you need to select the default TLS Context (Index #0, which is named "default"). If you are securing management through an Additional Management Interface, then select the TLS Context that you assigned the Additional Management Interface (in Configuring Additional Management Interfaces on page 46).

4. **(TLS Mutual Authentication Only)** Click the **Import** button, and then select the certificate file that was issued by the CA and which you want to import into the device's Trusted Root Certificates store.
5. Reset the device with a save-to-flash for your settings to take effect.

When a user connects to the secured Web interface of the device:

- If the user has a client certificate from a CA that is listed in the device's Trusted Root Certificate file, the connection is accepted and the user is prompted for the login password.
- If both the CA certificate and the client certificate appear in the Trusted Root Certificate file, the user is not prompted for a password. Therefore, this provides a single-sign-on experience; authentication is performed using the X.509 digital signature.
- If the user does not have a client certificate from a listed CA or does not have a client certificate, connection is rejected.

- The process of installing a client certificate on your PC is beyond the scope of this document. For more information, refer to your operating system documentation and consult with your security administrator.
- The root certificate can also be loaded through the device's Auto-Update mechanism, by using the [HTTPSRootFileName] parameter.
- You can enable the device to check whether a peer's certificate has been revoked by an OCSP server per TLS Context (see Configuring TLS Certificate Contexts).

Enabling CSRF Protection

The device's embedded Web server provides support for cross-site request forgery (CSRF) protection. CSRF prevents malicious exploits of a website, whereby unauthorized commands are transmitted from a user that the website trusts (i.e., authenticated user). Whenever a user opens (i.e., GET method) one of the device's Web pages, the device automatically generates a CSRF "token" (unique number). When the user performs actions (i.e., POST method) on that page (e.g., configures parameters), the token is included to verify that the authenticated user is the one performing the actions.

➢ **To enable CSRF protection:**

■ Load the device with an ini file containing the following parameter setting:

```
CSRFProtection = 1
```

Enabling DNS Rebinding Protection

The device provides protection against DNS rebinding attacks. DNS rebinding allows attackers to access and attack your device and internal network, by remapping hostname-to-IP address lookups. This may occur when you device use a hostname to access the device (see Configuring a Hostname for the Device on page 109) instead of its IP address. When the device blocks a DNS rebinding attack, it sends an HTTP 400 Bad Request response and logs the attack in Syslog.

➢ **To enable DNS rebinding protection:**

1. Open the Web Settings page (**Setup** menu > **Administration** tab > **Web & CLI** folder > **Web Settings**).
2. From the 'DNS Rebinding Protection' drop-down list, select **Enable**.

3. Click **Apply**.

Web Login Authentication using Smart Cards

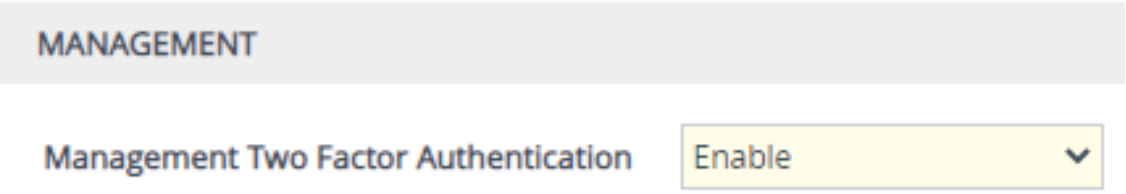
You can enable Web login authentication using certificates from a third-party, common access card (CAC) or smart card with user identification. When a user attempts to access the device through the Web browser (HTTPS), the device retrieves the Web user's login username (and other information, if required) from the CAC, and automatically displays it in the 'Username' field (read-only) on the Web Login screen. The user attempting to access the device is now only required to provide the login password.

Typically, a TLS connection is established between the CAC and the device's Web interface, and a RADIUS server is implemented to authenticate the password with the username. Therefore, this feature implements a two-factor authentication - what the user has (i.e., the physical card) and what the user knows (i.e., the login password).

For specific integration requirements for implementing a third-party smart card for Web login authentication, contact the sales representative of your purchased device.

➢ **To log in and enable Web login authentication using CAC:**

1. Open the Security Settings page (**Setup** menu > **IP Network** tab > **Security** folder > **Security Settings**).
2. From the 'Enable Management Two Factor Authentication' [EnableMgmtTwoFactorAuthentication] drop-down list, select **Enable**.



3. Insert the Common Access Card into your card reader.
4. Enter the password only. As some browsers may require a username, it's recommended to enter a username with an arbitrary value.

Configuring Web and Telnet Access List

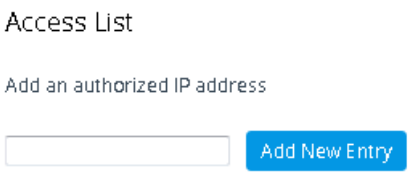
The Access List table lets you restrict access to the device's management interfaces (Web and CLI) by specifying up to 50 IP addresses (management clients) that are permitted to access the device. Access to the device's management interfaces from undefined IP addresses is denied (rejected with an HTTP 403 Forbidden response). If you don't specify any IP addresses, this security feature is inactive and the device can be accessed from any IP address.

The following procedure describes how to configure the Access List through the Web interface. You can also configure it through ini file [WebAccessList_x] or CLI (`configure system > mgmt-access-list`).

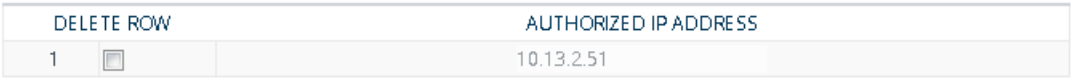
- Configure the IP address of the computer from which you are currently logged into the device as the first authorized IP address in the Access List. If you configure any other IP address, access from your computer will be immediately denied.
- If you configure network firewall rules in the Firewall table (see Configuring Firewall Rules), you must configure a firewall rule that permits traffic from IP addresses configured in the Access List table.

➢ **To add IP addresses to the Access List:**

1. Open the Access List table (**Setup** menu > **Administration** tab > **Web & CLI** folder > **Access List**).



2. In the 'Add an authorized IP address' field, configure an IP address, and then click **Add New Entry**; the IP address is added to the table.



If you have configured IP addresses in the Access List and you no longer want to restrict access to the management interface based on the Access List, delete all the IP addresses in the table, as described in the following procedure.

When deleting all the IP addresses in the Access List table, make sure that you delete the IP address of the computer from which you are currently logged into the device, **last**; otherwise, access from your computer will be immediately denied.

➢ **To delete an IP address from the Access List:**

1. Select the Delete Row check box corresponding to the IP address that you want to delete.
2. Click **Delete Selected Addresses**.

7 CLI-Based Management

This chapter provides an overview of the CLI-based management and provides configuration relating to CLI management.

- By default, CLI is disabled for security purposes.
- The CLI provides two access modes - Basic mode (basic commands) and Privileged mode (all commands). Access to these modes depends on management user level:
  - ✓ Monitor user level: Basic mode only
  - ✓ Administrator user level: Basic mode only
  - ✓ Security Administrator user level: Basic and Privileged modes
  - ✓ Master user level: Basic and Privileged modes
- For a description of the CLI commands, refer to the CLI Reference Guide.

Enabling CLI

By default, access to the device's CLI through Telnet and SSH is disabled. This section describes how to enable these protocols.

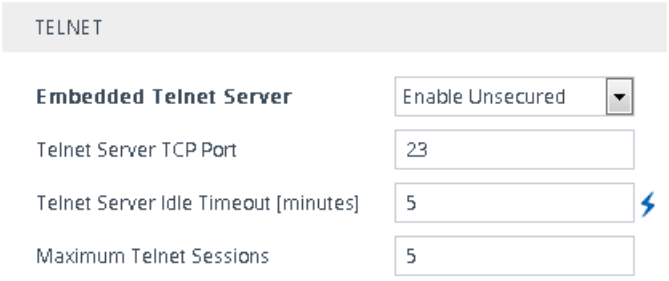
Enabling Telnet for CLI

The device provides an embedded Telnet server, which allows you to access its CLI from a remote Telnet client using the Telnet application protocol. By default, the Telnet server is enabled, but for unsecured Telnet connections whereby information is transmitted in clear text. Optionally, you can disable Telnet connectivity, or enable secured Telnet connections. If you enable secured Telnet connectivity, the device uses the TLS security protocol, whereby information is transmitted encrypted. For TLS, the device uses the TLS settings of the TLS Context at Index #0 ("default"). A special Telnet client is required on your PC to connect to the Telnet interface over the TLS connection, for example, C-Kermit for UNIX and Kermit-95 for Windows. For more information on TLS, see Configuring TLS Certificates on page 158.

For security, some organizations require the display of a proprietary notice upon starting a Telnet session. To configure such a message, see Creating a Login Welcome Message.

➢ **To enable Telnet:**

1. Open the CLI Settings page (**Setup** menu > **Administration** tab > **Web & CLI** folder > **CLI Settings**).



2. From the 'Embedded Telnet Server' drop-down list, select **Enable Unsecured** or **Enable Secured** (i.e, TLS) to enable Telnet.
3. In the 'Telnet Server TCP Port' field, enter the port number of the embedded Telnet server.
4. In the 'Telnet Server Idle Timeout' field, enter the duration of inactivity in the Telnet session after which the session automatically ends.
5. Click **Apply**, and then reset the device with a save-to-flash for your settings to take effect.

For a detailed description of the Telnet parameters, see Telnet and CLI Parameters.

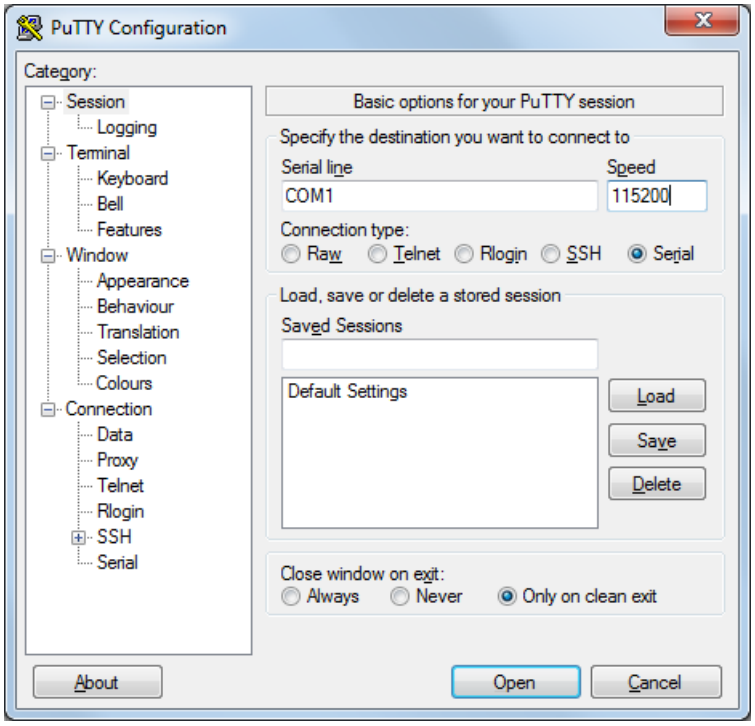
Enabling SSH with RSA Public Key for CLI

Unless configured for TLS, Telnet is not secure as it requires passwords to be transmitted in clear text. To overcome this, you can use Secure SHell (SSH) which is the de-facto standard for secure CLI. SSH 2.0 is a protocol built above TCP providing methods for key exchange, authentication, encryption, and authorization. SSH requires appropriate client software for the management PC. Most Linux distributions have OpenSSH pre-installed; Windows-based PCs require an SSH client software such as PuTTY. By default, SSH uses the same username and password as the device's Telnet and Web server. SSH supports 1024/2048-bit RSA public keys, providing carrier-grade security.

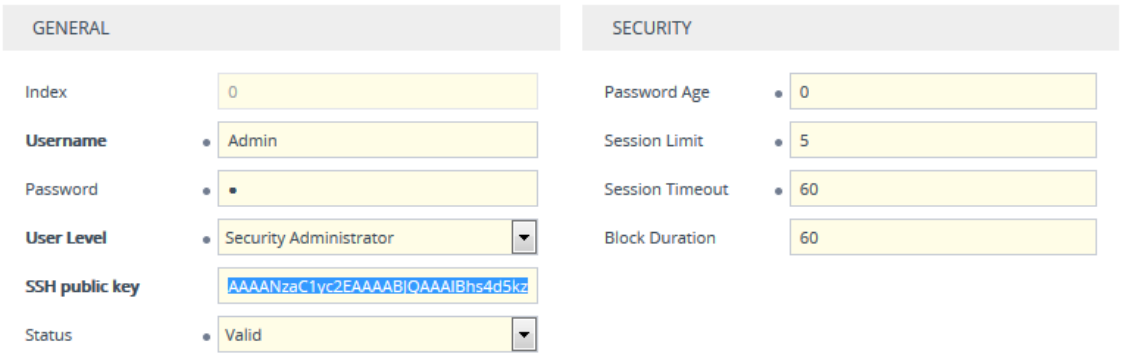
Follow the instructions below to configure the device with an administrator RSA key as a means of strong authentication.

➢ **To enable SSH and configure RSA public keys for Windows (using PuTTY SSH software):**

1. Start the PuTTY Key Generator program, and then do the following:
   a. Under the Parameters group, do the following:
      - Select the **SSH-2 RSA** option.
      - In the 'Number of bits in a generated key' field, enter "1024" bits.
   b. Under the Actions group, click **Generate** and then follow the on-screen instructions.
   c. Under the Actions group, click **Save private key** to save the new private key to a file (*.ppk) on your PC.
   d. Under the Key group, select and copy the displayed encoded text (public key) between "ssh-rsa" and "rsa-key-….", as shown in the example below:



2. Open the Local Users table (see Configuring Management User Accounts), and then for the required user, paste the public key that you copied in Step 1.d into the 'SSH Public Key' field, as shown below:



> The public key cannot be configured with wide characters.

3. On the CLI Settings page, do the following:

   a. From the 'Enable SSH Server' drop-down list, select **Enable**.

   b. For additional security, you can configure the 'Public Key' field to **Enable**. This ensures that SSH access is only possible by using the RSA key and not by username and password.

   c. Configure the other SSH parameters as required. For a description of these parameters, see SSH Parameters.

d. Click **Apply**.

4. Start the PuTTY Configuration program, and then do the following:
   a. In the 'Category' tree, drill down to **Connection**, then **SSH**, and then **Auth**; the 'Options controlling SSH authentication' pane appears.
   b. Under the 'Authentication parameters' group, click **Browse** and then locate the private key file that you created and saved in Step 4.
5. Connect to the device with SSH using the username "Admin"; RSA key negotiation occurs automatically and no password is required.

➢ **To configure RSA public keys for Linux (using OpenSSH 4.3):**

1. Run the following command to create a new key in the admin.key file and to save the public portion to the admin.key.pub file:

```
ssh-keygen -f admin.key -N "" -b 1024
```

2. Open the admin.key.pub file, and then copy the encoded string from "ssh-rsa" to the white space.
3. Open the Local Users table (see Configuring Management User Accounts), and then for the required user, paste the public key that you copied in Step 2 into the 'SSH Public Key' field.
4. Connect to the device with SSH, using the following command (where *xx.xx.xx.xx* is the device's IP address):

```
ssh -i admin.key xx.xx.xx.xx
```

RSA-key negotiation occurs automatically and no password is required.

Configuring Maximum Telnet/SSH Sessions

You can configure the maximum number of concurrent Telnet and SSH sessions permitted on the device.

- Before changing the setting, make sure that not more than the number of sessions that you want to configure are currently active; otherwise, the new setting will not take effect.
- The device supports up to five concurrent Telnet and SSH sessions.

➢ **To configure the maximum number of concurrent Telnet and SSH sessions:**

1. Open the CLI Settings page (**Setup** menu > **Administration** tab > **Web & CLI** folder > **CLI Settings**).

2. **For Telnet:** Under the **Telnet** group, in the 'Maximum Telnet Sessions' field, enter the maximum number of concurrent sessions.
3. **For SSH:** Under the **SSH** group, in the 'Maximum SSH Sessions' field, enter the maximum number of concurrent sessions.
4. Click **Apply**.

Establishing a CLI Session

You can access the device's CLI using any of the following methods:

- **RS-232:** The device can be accessed through its RS-232 serial port, by connecting a VT100 terminal to it or using a terminal emulation program (e.g., HyperTerminal) with a PC. For connecting to the CLI through RS-232, see CLI.
- **Secure SHell (SSH):** The device can be accessed through its Ethernet interface by the SSH protocol using SSH client software. A popular and freeware SSH client software is Putty, which can be downloaded from http://www.chiark.greenend.org.uk/~sgtatham/putty/download.html.
- **Telnet:** The device can be accessed through its Ethernet interface by the Telnet protocol using Telnet client software.

The following procedure describes how to access the CLI through Telnet/SSH.

The CLI login credentials are the same as all the device's other management interfaces (such as Web interface). The default username and password is "Admin" and "Admin" (case- sensitive), respectively. To configure login credentials and management user accounts, see Configuring Management User Accounts.

➢ To establish a CLI session through Telnet or SSH:

1. Connect the device to the network.
2. Establish a Telnet or SSH session using the device's OAMP IP address.
3. Log in to the session using the username and password assigned to the Admin user of the Web interface:
   a. At the Username prompt, type the username, and then press Enter:

      ```
      Username: Admin
      ```

   b. At the Password prompt, type the password, and then press Enter:

      ```
      Password: Admin
      ```

   c. At the prompt, type the following, and then press Enter:

```
> enable
```

d. At the prompt, type the password again, and then press Enter:

```
Password: Admin
```

Viewing Current CLI Sessions

You can view users that are currently logged in to the device's CLI. This applies to users logged in to the CLI through RS-232 (console), Telnet, or SSH. For each logged-in user, the following is displayed: the type of interface (console, Telnet, or SSH), username, remote IP address from where the user logged in, and the duration (days and time) of the session. Each user is displayed with a unique index (session ID).

➢ **To view currently logged-in CLI users:**

1. Establish a CLI session with the device.
2. Run the following command:

```
# show users
[0] console    Admin     local          0d00h03m15s
[1] telnet     John      10.4.2.1       0d01h03m47s
[2]* ssh       Alex      192.168.121.234  12d00h02m34s
```

The current session from which this show command was run is displayed with an asterisk (*).

The device can display management sessions of up to 24 hours. After this time, the duration counter is reset.

Terminating a User's CLI Session

You can terminate users that are currently logged in to the device's CLI. This applies to users logged in to the CLI through RS-232 (console), Telnet, or SSH.

➢ **To terminate the CLI session of a specific CLI user:**

1. Establish a CLI session with the device.
2. Run the following command:

```
# clear user <session ID>
```

Where <*session ID*> is a unique identification of each currently logged in user. You can view the session ID by running the **show users** command (see Viewing Current CLI Sessions).

The session in which the command is run cannot be terminated.

Configuring Displayed Output Lines in CLI Terminal Window

You can configure the maximum number of lines (height) displayed in the terminal window for the output of CLI commands (Telnet and SSH). The number of displayed lines can be from 0 to 65,535, or determined by re-sizing the terminal window by mouse-dragging the window's border.

➢ **To specify the number of displayed output lines:**

1. Establish a CLI session with the device.
2. Access the System menu:

```
# configure system
```

3. At the prompt, type the following command:

```
(config-system)# cli-terminal
```

4. At the prompt, type the following command:

```
<cli-terminal># window-height [0-65535]
```

If window-height is set to 0, the entire command output is displayed. In other words, even if the output extends beyond the visible terminal window length, the --MORE-- prompt is not displayed.

➢ **To configure the number of displayed output lines by dragging terminal window:**

1. Establish a CLI session with the device.
2. Access the System menu:

```
# configure system
```

3. At the prompt, type the following command:

```
(config-system)# cli-terminal
```

4. At the prompt, type the following command:

```
<cli-terminal># window-height automatic
```

When this mode is configured, each time you change the height of the terminal window using your mouse (i.e., dragging one of the window's borders or corners), the number of displayed output command lines is changed accordingly.

Idle CLI Session Timeout for RS-232 Connections

If you have established a CLI session (successfully logged in) with the device through an RS-232 serial interface and you don't perform any actions in the CLI session for five minutes, the device automatically logs you out the session. In such a scenario, you need to log in to the CLI again if you want to continue using the CLI. This idle session timeout is not configurable.

Configuring Password Display in CLI

You can enable the device to display passwords in the CLI's `show running-config` output in encrypted (obscured) format instead of in plain text. When passwords are displayed encrypted, the word "obscured" appears after the password.

➢ **To enable obscured password display in CLI:**

```
(config-system)# cli-settings
(cli-settings)# password-obscurity on
```

Below shows two examples of password display (obscured and plain text) in the `show running-config` output for a password configured for a Remote Web Service:

- Password displayed in encrypted (obscured) format:

```
rest-password 8ZybmJHExMTM obscured
```

- Password displayed in plain text:

```
rest-password John1234
```

8 SNMP-Based Management

The device provides an embedded SNMP agent that lets you manage it using AudioCodes One Voice Operations Center (OVOC) or a third-party SNMP manager. The SNMP agent supports standard and proprietary Management Information Base (MIBs). All supported MIB files are supplied to customers as part of the release. The SNMP agent can send unsolicited SNMP trap events to the SNMP manager.

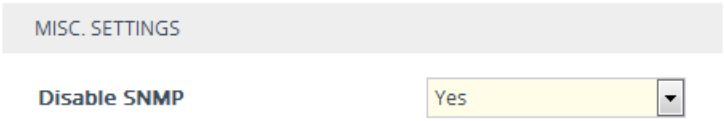
- By default, SNMP-based management is enabled.
- For more information on the device's SNMP support such as SNMP trap alarms and events, refer to the *SNMP Reference Guide*.
- For more information on OVOC, refer to the *OVOC User's Manual*.

Disabling SNMP

By default, SNMP is enabled. However, you can disable it as described in the following procedure.

➢ **To disable SNMP:**

1. Open the SNMP Community Settings page (**Setup** menu > **Administration** tab > **SNMP** folder > **SNMP Community Settings**).
2. From the 'Disable SNMP' drop-down list (DisableSNMP parameter), select **Yes**:



3. Click **Apply**, and then reset the device with a save-to-flash for your settings to take effect.

Configuring SNMP Community Strings

SNMP community strings determine the access privileges (read-only and read-write) of SNMP clients with the device's SNMP agent. You can configure up to five read-only SNMP community strings and up to five read-write SNMP community strings. The device's SNMP agent accepts SNMP Get (read-only) and Set (read-write) requests only if the correct community string is used in the request.

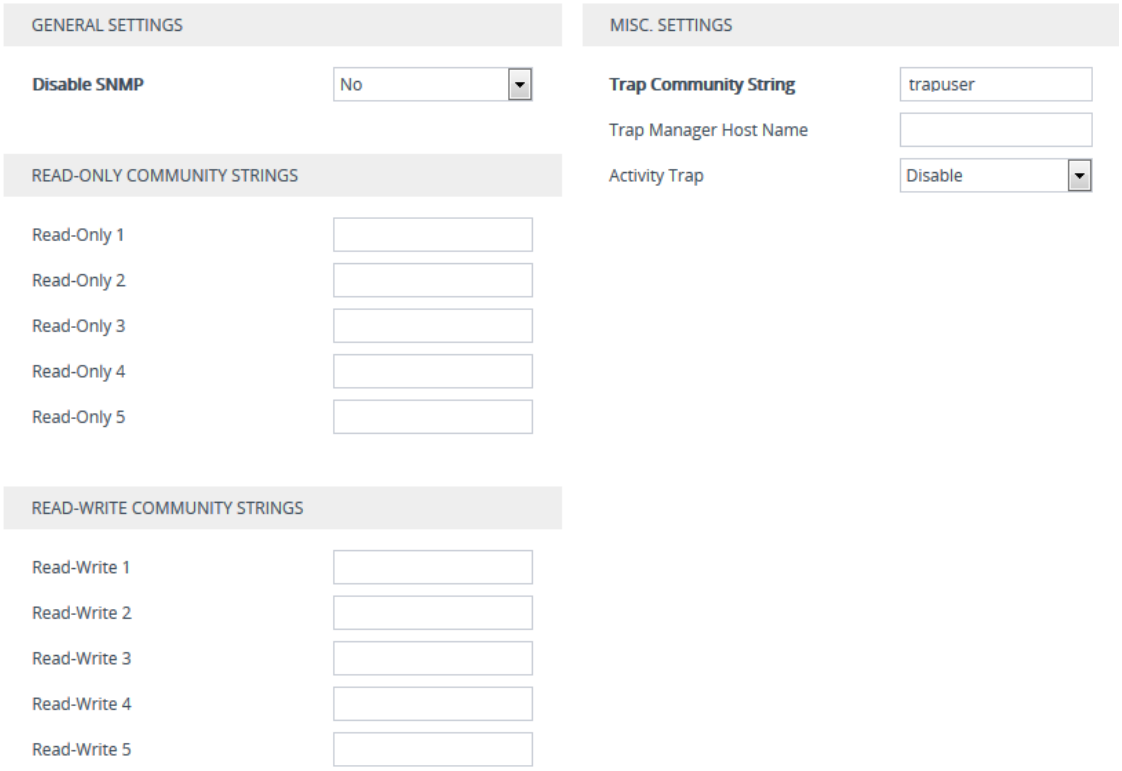
You can also configure a unique password-like community string used for sending SNMP traps. The device sends the traps with the community string.

- SNMP community strings are applicable only to SNMPv1 and SNMPv2c. SNMPv3 uses username-password authentication along with an encryption key (see Configuring SNMP V3 Users).
- If you configure SNMPv3 users (see Configuring SNMP V3 Users on page 84), the device ignores all SNMP requests (Get and Set operations) from SNMPv2 users (sends the authenticationFailure trap).
- The read-only community strings must be different to the read-write community strings.
- You can enhance security by configuring Trusted Managers (see Configuring SNMP Trusted Managers). A Trusted Manager is an IP address from which the SNMP agent accepts Get and Set requests.

For detailed descriptions of the SNMP parameters, see SNMP Parameters.

➢ To configure SNMP community strings:

1. Open the SNMP Community Settings page (**Setup** menu > **Administration** tab > **SNMP** folder > **SNMP Community Settings**).
2. Configure SNMP community strings for access privileges:



- Under the **Read-Only Community Strings** group, configure read-only community strings (see the table below).
- Under the **Read-Write Community Strings** group, configure read-write community strings (see the table below).

3. Configure a community string for SNMP traps: Under the **Misc. Settings** group, in the 'Trap Community String' field, configure a community string (see the table below).

4. Click **Apply**, and then reset the device with a save-to-flash for your settings to take effect.

To delete a community string, delete the configured string, click **Apply**., and then reset the device with a save-to-flash for your settings to take effect.

Table 8-1: SNMP Community String Parameter Descriptions

Parameter	Description
'Read-Only Community Strings' `configure system > snmp settings > ro-community-string` [SNMPReadOnlyCommunityStringsPassword_x]	Defines read-only SNMP community strings. Up to five read-only community strings can be configured. The valid value is a string of up to 30 characters that can include only the following: ■ Upper- and lower-case letters (a to z, and A to Z) ■ Numbers (0 to 9) ■ Hyphen (-) ■ Underline (_) For example, "Public-comm_string1". The default is "public". **Note:** ■ The parameter cannot be configured with wide characters. ■ The read-only community strings must be different to the read-write community strings. ■ For ini file configuration, *x* is 0 for the 'Read-Only 1' parameter.
'Read-Write Community Strings' `configure system > snmp settings > rw-community-string` [SNMPReadWriteCommunityStringsPassword_x]	Defines read-write SNMP community strings. Up to five read-write community strings can be configured. The valid value is a string of up to 30 characters that can include only the following:

Parameter	Description
	■ Upper- and lower-case letters (a to z, and A to Z) ■ Numbers (0 to 9) ■ Hyphen (-) ■ Underline (_) For example, "Private-comm_string1". The default is "private". ■ The parameter cannot be configured with wide characters. ■ The read-write community strings must be different to the read-only community strings. ■ For ini file configuration, *x* is 0 for the 'Read-Write 1' parameter.
'Trap Community String' `configure system > snmp trap > community-string` [SNMPTrapCommunityStringPassword]	Defines the community string for SNMP traps. The valid value is a string of up to 30 characters that can include only the following: ■ Upper- and lower-case letters (a to z, and A to Z) ■ Numbers (0 to 9) ■ Hyphen (-) ■ Underline (_) For example, "Trap-comm_string1". The default is "trapuser". **Note:** The parameter cannot be configured with wide characters.

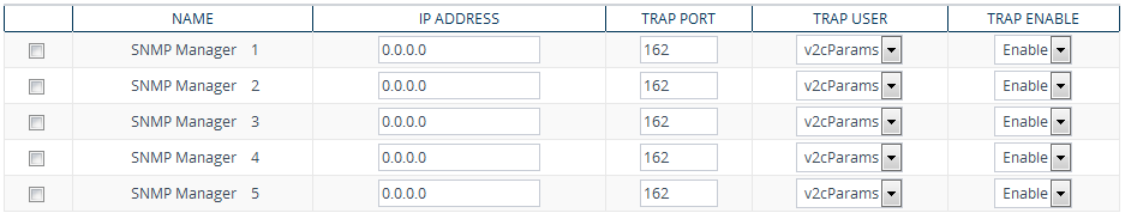
Configuring SNMP Trap Destinations with IP Addresses

The SNMP Trap Destinations table lets you configure up to five SNMP trap managers to receive traps sent by the device. The SNMP manager is defined by IP address (IPv4) and port. You can associate a trap destination with SNMPv2 users and specific SNMPv3 users. Associating a trap destination with SNMPv3 users sends encrypted and authenticated traps to the SNMPv3 destination. By default, traps are sent unencrypted using SNMPv2.

The following procedure describes how to configure SNMP trap destinations through the Web interface. You can also configure it through ini file [SNMPManager] or CLI (`configure system > snmp trap-destination`).

➢ **To configure SNMP trap destinations:**

1. Open the SNMP Trap Destinations table (**Setup** menu > **Administration** tab > **SNMP** folder > **SNMP Trap Destinations**).



	NAME	IP ADDRESS	TRAP PORT	TRAP USER	TRAP ENABLE
☐	SNMP Manager 1	0.0.0.0	162	v2cParams	Enable
☐	SNMP Manager 2	0.0.0.0	162	v2cParams	Enable
☐	SNMP Manager 3	0.0.0.0	162	v2cParams	Enable
☐	SNMP Manager 4	0.0.0.0	162	v2cParams	Enable
☐	SNMP Manager 5	0.0.0.0	162	v2cParams	Enable

2. Configure the SNMP trap manager according to the table below.
3. Select the check boxes corresponding to the configured SNMP managers that you want to enable.
4. Click **Apply**.

- Rows whose corresponding check boxes are cleared revert to default settings when you click **Apply**.
- To enable the sending of the trap event, acPerformanceMonitoringThresholdCrossing, which is sent whenever a threshold (high or low) of a performance monitored SNMP MIB object is crossed, configure the ini file parameter [PM_EnableThresholdAlarms] to [1]. Once enabled, you can change its default low and high threshold values. For more information, see the *SNMP Reference Guide for Gateways-SBCs-MSBRs*.
- Instead of configuring SNMP trap managers with an IP address in dotted-decimal notation, you can configure a single SNMP trap manager with an FQDN (see Configuring an SNMP Trap Destination with FQDN.

Table 8-2: SNMP Trap Destinations Table Parameters Description

Parameter	Description
(check box) [SNMPManagerIsUsed_x]	Enables the SNMP manager to receive traps and checks the validity of the configured destination (IP address and port number). ■ [0] (check box cleared) = (Default) Disables SNMP manager ■ [1] (check box selected) = Enables SNMP manager
'IP Address'	Defines the IP address of the remote host used

Parameter	Description
[SNMPManagerTableIP_x]	as the SNMP manager. The device sends its SNMP traps to this IP address. The valid value is an IPv4 address (in dotted-decimal notation, e.g., 108.10.1.255). **Note:** If you are using a WebSocket tunnel connection between the device and OVOC, then configure the parameter to the IP address mentioned in Configuring WebSocket Tunnel with OVOC on page 91
'Trap Port' [SNMPManagerTrapPort_x]	Defines the port number of the remote SNMP manager. The device sends SNMP traps to this port. The valid value range is 100 to 4000. The default is 162.
'Trap User' [SNMPManagerTrapUser]	Associates a trap user (SNMPv2 or SNMPv3) with the trap destination. This determines the trap format, authentication level, and encryption level. ■ **v2cParams** = (Default) SNMPv2 user community string ■ SNMPv3 user configured in Configuring SNMP V3 Users
'Trap Enable' [SNMPManagerTrapSendingEnable_x]	Activates the sending of traps to the SNMP Manager. ■ [0] **Disable** ■ [1] **Enable** (default)

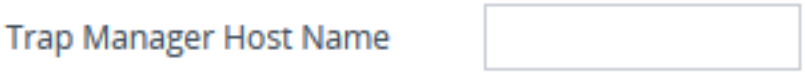
Configuring an SNMP Trap Destination with FQDN

Instead of configuring SNMP trap destinations (managers) with IP addresses in the SNMP Trap Destinations table (see Configuring SNMP Trap Destination with IP Addresses), you can configure a single SNMP trap manager with an FQDN (e.g., mngr.corp.mycompany.com). The device sends the traps to the DNS-resolved IP address. The resolved IP address replaces the IP address of the last row (SNMP Manager 5) in the SNMP Trap Destinations table (and the last trap manager entry in the snmpTargetAddrTable in the snmpTargetMIB).

- If you configure an FQDN for an SNMP trap manager, the device ignores your configuration in the SNMP Trap Destinations table.
- If you configure an FQDN for an SNMP trap manager, only one SNMP trap manager can be configured.
- To resolve the FQDN into an IP address, the device uses the DNS server that is configured in the IP Interfaces table for the IP Interface whose 'Application Type' is **OAMP**.

➢ **To configure an SNMP trap destination with an FQDN:**

1. Open the SNMP Community Settings page (**Setup** menu > **Administration** tab > **SNMP** folder > **SNMP Community Settings**).
2. Under the **Misc. Settings** group, in the 'Trap Manager Host Name' field [SNMPTrapManagerHostName], enter the FQDN.



3. Click **Apply**.

Configuring SNMP Trusted Managers

The SNMP Trusted Managers table lets you configure up to five SNMP Trusted Managers. By default, the SNMP agent accepts SNMP Get and Set requests from any IP address as long as the correct community string is used in the request (see Configuring SNMP Community Strings). You can enhance security by configuring Trusted Managers, which is an IP address (IPv4) from which the device's SNMP agent accepts and processes SNMP requests. If no SNMP Trusted Manager is configured, any SNMP manager can access the device (as long as the community string is correct).

The following procedure describes how to configure SNMP Trusted Managers through the Web interface. You can also configure it through ini file [SNMPTrustedMgr_x] or CLI (`configure system > snmp settings > trusted-managers`).

➢ **To configure SNMP Trusted Managers:**

1. Open the SNMP Trusted Managers table (**Setup** menu > **Administration** tab > **SNMP** folder > **SNMP Trusted Managers**).

DELETE	TRUSTED MANAGERS IP ADDRESS	
☐	SNMP Trusted Manager 1	0.0.0.0
☐	SNMP Trusted Manager 2	0.0.0.0
☐	SNMP Trusted Manager 3	0.0.0.0
☐	SNMP Trusted Manager 4	0.0.0.0
☐	SNMP Trusted Manager 5	0.0.0.0

2. Configure an IP address (IPv4) for one or more SNMP Trusted Managers.
3. Select the check boxes corresponding to the configured SNMP Trusted Managers that you want to enable.
4. Click **Apply**, and then reset the device with a save-to-flash for your settings to take effect.

Enabling SNMP Traps for Web Activity

You can enable the device to send SNMP traps to notify of management users' activities in the Web interface. A trap is sent each time an activity is done by a user. To configure the types of Web activities that you want reported, see Configuring Reporting of Management User Activities.

➢ **To enable traps to SNMP manager for Web activity:**

1. Open the SNMP Community Settings page (**Setup** menu > **Administration** tab > **SNMP** folder > **SNMP Community Settings**).
2. Under the **Misc. Settings** group, from the 'Activity Trap' drop-down list (EnableActivityTrap), select **Enable**.

3. Click **Apply**.

Configuring SNMP V3 Users

The SNMPv3 Users table lets you configure up to 10 SNMP v3 users for authentication and privacy.

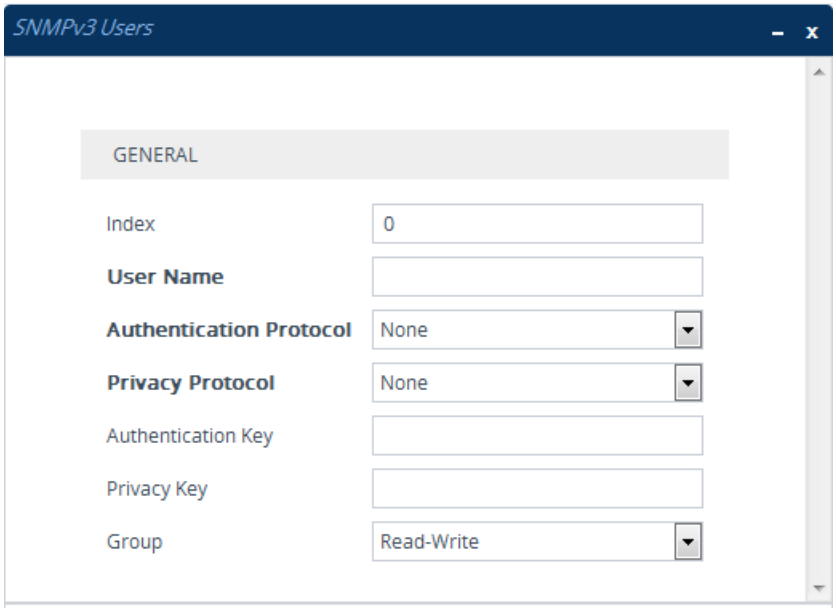
The following procedure describes how to configure SNMP v3 users through the Web interface. You can also configure it through ini file [SNMPUsers] or CLI (`configure system > snmp v3-users`).

- If you delete a user that is associated with a trap destination (see Configuring SNMP Trap Destinations with IP Addresses), the trap destination becomes disabled and the trap user reverts to default (i.e., SNMPv2).
- If you configure an SNMPv3 user(s), the device ignores all SNMP requests (Get and Set operations) from SNMPv2 users (sends the authenticationFailure trap).
- If you want to use the same SNMPv3 Users table configuration for another device, before uploading this device's configuration file (.ini) to the other device, you **must** edit the file so that the passwords ('Authentication Key' and 'Privacy Key' parameters) are in plain text.

➢ **To configure an SNMP v3 user:**

1. Open the SNMPv3 Users table (**Setup** menu > **Administration** tab > **SNMP** folder > **SNMP V3 Users**).

2. Click **New**; the following dialog box appears:



3. Configure the SNMP V3 parameters according to the table below.
4. Click **Apply**.

Table 8-3: SNMPv3 Users Table Parameters Description

Parameter	Description
'Index' [SNMPUsers_ Index]	Defines an index number for the new table row. **Note:** Each row must be configured with a unique index.
'User Name' `username` [SNMPUsers_ Username]	Name of the SNMP v3 user. The name must be unique.
'Authentication Protocol' `auth-protocol` [SNMPUsers_ AuthProtocol]	Authentication protocol of the SNMP v3 user. ■ [0] **None** (default) ■ [1] **MD5** ■ [2] **SHA-1**
'Privacy Protocol' `priv-protocol` [SNMPUsers_ PrivProtocol]	Privacy protocol of the SNMP v3 user. ■ [0] **None** (default) ■ [1] **DES** ■ [2] **3DES**

Parameter	Description
	■ [3] **AES-128**
'Authentication Key' `auth-key` [SNMPUsers_AuthKey]	Authentication key. Keys can be entered in the form of a text password or long hex string. Keys are always persisted as long hex strings and keys are localized. The value must be at least six characters (preferably 8 characters). **Note:** The parameter cannot be configured with wide characters.
'Privacy Key' `priv-key` [SNMPUsers_PrivKey]	Privacy key. Keys can be entered in the form of a text password or long hex string. Keys are always persisted as long hex strings and keys are localized.
'Group' `group` [SNMPUsers_Group]	The group with which the SNMP v3 user is associated. ■ [0] **Read-Only** ■ [1] **Read-Write** (default) ■ [2] **Trap** **Note:** All groups can be used to send traps.

Customizing SNMP Alarm Severity

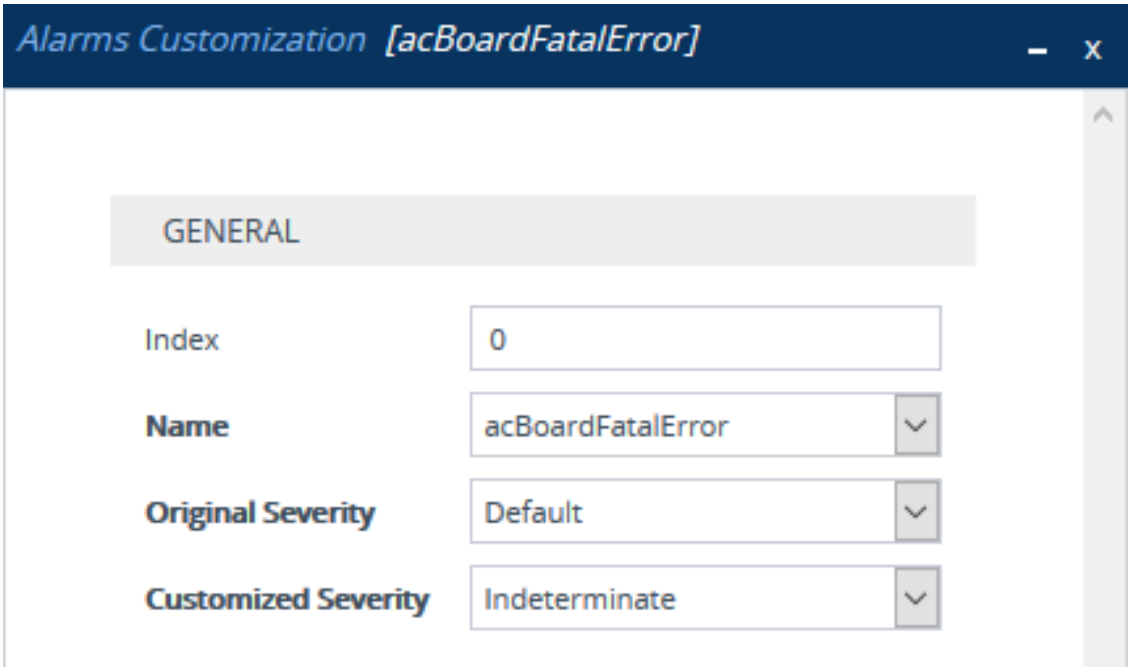
The Alarms Customization table lets you configure up to 150 Alarm Customization rules. The table allows you to customize the severity levels of the device's SNMP trap alarms. The table also allows you to disable (*suppress*) an alarm all together or a specific alarm severity. For example, by default, when an alarm cannot be entered in the Active Alarms table due to it being full, the device sends the acActiveAlarmTableOverflow alarm with a severity level of Major. By using this table, you can customize this alarm condition and change the severity level to Warning, for example.

- If you have customized an alarm that has subsequently been sent by the device and you then delete the rule when the alarm is still active, the device doesn't send the alarm again for that instance. For example, assume that you customize the severity of the acBoardEthernetLinkAlarm alarm to **Warning** and the Ethernet cable is subsequently disconnected. If you then delete the rule while this condition still exists (i.e., cable still disconnected), the device does not re-send the acBoardEthernetLinkAlarm alarm (with the default severity level -- Major or Minor).
- If you configure multiple Alarm Customization rules for the **same** alarm, out of all these same rules the device applies only the rule that you configured first (i.e., listed highest in the table -- with lowest Index) and ignores the others.
- After an HA switchover, all disabled (**Suppressed**) alarms are restored (not suppressed).

The following procedure describes how to customize alarm severity levels through the Web interface. You can also configure it through ini file [AlarmSeverity] or CLI (`configure system > snmp alarm-customization`).

➢ **To customize SNMP alarm severity levels:**

1. Open the Alarms Customization table (**Setup** menu > **Administration** tab > **SNMP** folder > **Alarm Customization**).



2. Configure a rule according to the parameters described in the table below.
3. Click **Apply**, and then reset the device with a save-to-flash for your settings to take effect.

Table 8-4: Alarms Customization Parameter Descriptions

Parameter	Description
Index [AlarmSeverity_Index]	Defines an index number for the new table row. **Note:** Each row must be configured with a unique index.
Name `name` [AlarmSeverity_Name]	Defines the SNMP alarm that you want to customize. **Note:** The CLI and ini file use the last digits of the alarm's OID as the name. For example, configure the parameter to "12" for the acActiveAlarmTableOverflow alarm (OID is 1.3.6.1.4.15003.9.10.1.21.2.0.**12**). For alarm OIDs, refer to the *SNMP Reference Guide for Gateways-SBCs-MSBRs*.
Original Severity `alarm-original-severity` [AlarmSeverity_ OriginalSeverity]	Defines the original severity level of the alarm, according to the MIB. ■ [0] **Default** = (Default) All supported severity levels of the alarm. If you select this option, the alarm and its' severity depends on the 'Original Severity' parameter: ✓ If configured to **Suppressed**, the device doesn't send the alarm at all. ✓ If configured to any value other than **Suppressed**, the device always sends the alarm with the configured severity (regardless of condition). ■ [1] **Indeterminate** ■ [2] **Warning** ■ [3] **Minor** ■ [4] **Major** ■ [5] **Critical**
Customized Severity `alarm-customized-severity` [AlarmSeverity_ CustomizedSeverity]	Defines the new (customized) severity of the alarm. This severity replaces the alarm's original severity that you specified in the 'Original Severity' parameter. For example, if you want to change the severity of the acCertificateExpiryAlarm alarm from Minor to Major, then configure the 'Original Severity' parameter to **Minor** and the 'Customized Severity' parameter to **Major**. ■ [0] **Suppressed** = Disables (suppresses) the alarm or a specified severity, depending on the 'Original Severity' parameter: ✓ To suppress an alarm: Configure the 'Original

Parameter	Description
	Severity' parameter to **Default**, or if the alarm has only one severity level, configure the 'Original Severity' parameter to this severity. For example, as the acBoardConfigurationError alarm is only sent with Critical severity, configure the 'Original Severity' parameter to **Critical**. ✓ To suppress the sending of a specific alarm severity: If the alarm has multiple severity levels (based on conditions), configure the 'Original Severity' parameter to the severity that you don't want the device to send. For example, if you don't want the device to send the acProxyConnectionLost alarm when its' severity is Minor, configure the 'Original Severity' parameter to **Minor**. ■ [1] **Indeterminate** (default) ■ [2] **Warning** ■ [3] **Minor** ■ [4] **Major** ■ [5] **Critical**

Configuring SNMP for OVOC Connectivity

Connection between the device and OVOC is through SNMP. Once connected, the device can send SNMP traps to OVOC, and OVOC can perform various operations on the device such as maintenance actions, and fault and performance management.

- Make sure that the SNMP settings on the device and on OVOC are **identical**.
- OVOC uses the following default settings:
  - ✓ Trap port: **162** (configured in the SNMP Trap Destinations table, as described below).
  - ✓ SNMPv2: **public** for the read-community string, **private** for read-write community string, and **trapuser** for the trap community string (configured on the SNMP Community Settings page, as described below).
  - ✓ SNMPv3: **OVOCUser** for user name; **SHA-1** for authentication protocol; **AES-128** for privacy protocol; **123456789** for the 'Authentication Key' and 'Privacy Key' password (configured in the SNMPv3 Users table, as described below).
- If the device is located behind NAT and you have added it to OVOC by serial number or by auto-detection, you also need to configure (through ini file) the device to send NAT keep-alive traps to the OVOC port to keep the NAT pinhole open for SNMP messages sent from OVOC to the device:
  - ✓ [SendKeepAliveTrap] = [**1**]
  - ✓ [KeepAliveTrapPort] = [**1161**]
  - ✓ [NatBindingDefaultTimeout] = [**30**]
- When the device operates in High-Availability (HA) mode and it sends alarms to OVOC, the name of the device as configured by the 'HA Device Name' parameter is displayed at the beginning of the alarm description in OVOC, for example, " (SBCSITE01): Proxy lost. looking for another proxy". However, the name is not displayed for the alarms retrieved (from the device's Active Alarms table) when OVOC **initially** connects to the device.

➢ **To configure SNMP for device-OVOC connectivity:**

1. Make sure that SNMP is enabled, which it is by default (see Disabling SNMP on page 77).
2. Configure the local SNMP port (for Get/Set commands) on the device to 161, using the [SNMPPort] parameter.
3. Configure an SNMPv2 or SNMPv3 user:
   - **For SNMPv2 user:**
     i. Open the SNMP Community Settings page (Configuring SNMP Community Strings on page 77).
     ii. In the 'Read-Only 1' parameter [SNMPReadCommunity), configure the SNMP read-only community string.
     iii. In the 'Read-Write 1' parameter [SNMPWriteCommunity], configure the SNMP read-write community string.
     iv. In the 'Trap Community String' parameter [SNMPTrapCommunityStringPassword], configure the community string for SNMP traps.
   - **For SNMPv3 users:**
     i. Open the SNMPv3 Users table (see Configuring SNMP V3 Users on page 84).
     ii. In the 'User Name' parameter, configure the name of the SNMP v3 user.

iii. From the 'Authentication Protocol' drop-down list, select the authentication protocol.

iv. From the 'Privacy Protocol' drop-down list, select the privacy protocol.

v. In the 'Authentication Key' and 'Privacy Key' parameters, configure the password.

4. Configure the device to send its traps to OVOC (acting as an SNMP Manager), in the SNMP Trap Destinations table (see Configuring SNMP Trap Destinations with IP Addresses on page 80):

   a. In the 'IP Address' parameter, configure the OVOC IP address.

   b. In the 'Trap Port' parameter, configure the OVOC port.

   c. From the 'Trap User' drop-down list, select a trap user (SNMPv2 or SNMPv3) for this trap destination.

   d. From the 'Trap Enable' drop-down list, select **Enable**.

   Below shows an example where OVOC is configured as an SNMP Manager with IP address:port 172.17.118.219:162 and using an SNMPv3 user:

> If the OVOC address is an FQDN, instead of configuring the SNMP Manager (OVOC) above with an IP address, you can configure a single SNMP trap manager with an FQDN, as described in Configuring an SNMP Trap Destination with FQDN.

5. If the device is located behind NAT and you have added the device to OVOC by its serial number or using auto-detection, you also need to configure (through ini file) the device to send NAT keep-alive traps to the OVOC port to keep the NAT pinhole open for SNMP messages sent from OVOC to the device:

   a. Enable the sending of NAT keep-alive traps to OVOC, by configuring the [SendKeepAliveTrap] parameter to [1].

   b. Define the OVOC port to where the device sends the NAT keep-alive traps, by using the [KeepAliveTrapPort] parameter.

   c. Define the interval between each sent NAT keep-alive trap, by using the [NatBindingDefaultTimeout] parameter.

6. Reset the device with a save-to-flash for your settings to take effect.

Configuring WebSocket Tunnel with OVOC

When OVOC is deployed in a public cloud environment (e.g., Amazon Web Services), it can manage devices that are located **behind NAT**, by implementing WebSocket tunneling (over HTTP/S). All communication and management traffic (e.g., HTTP-based file download, NTP, Syslog, debug recording, and SNMP) between the device and OVOC flows through this

WebSocket tunnel. In this tunneling application, the device is the WebSocket client and OVOC is the WebSocket server.

WebSocket tunnel has many advantages over the alternative method for connecting OVOC to the device when located behind NAT (refer to One Voice Operations Center IOM Manual for more information). It easily resolves NAT traversal problems and requires minimal amount of configuration, for example, there's no need for port forwarding nor firewall settings to allow certain traffic.

The WebSocket tunnel connection between the device and OVOC is secure (HTTPS). When the device initiates a WebSocket tunnel connection, it verifies that the TLS certificate presented by OVOC is signed by one of the CAs in the trusted root store of its default TLS Context (ID #0). The device authenticates itself with OVOC using a username and password. These must be the same credentials as configured on OVOC.

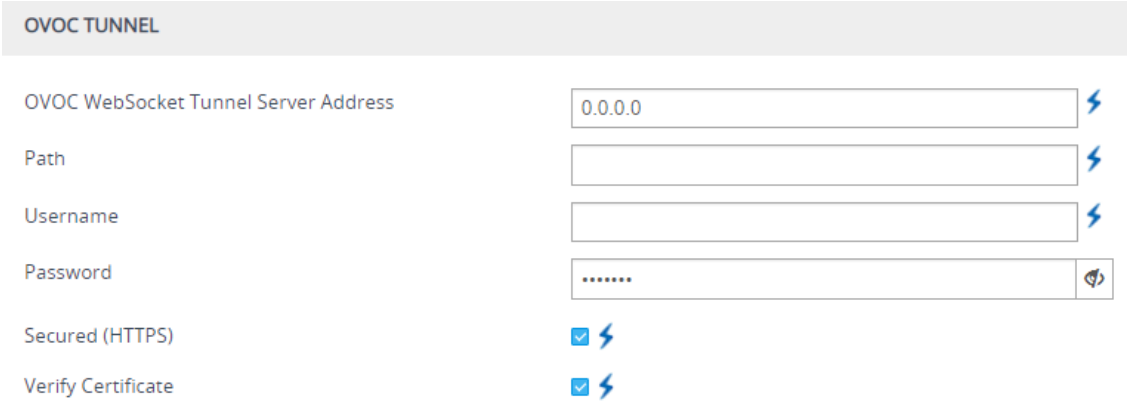
The device establishes the WebSocket connection through the OAMP IP interface. The device keeps the WebSocket tunnel connection open (i.e., persistent), allowing it to send and receive future management traffic through it. The connection only closes before the device (or OVOC) restarts.

- when is Microsoft Azure, Amazon , VMware, or Microsoft Hyper-V To check if its supported on additional cloud platforms, refer to the OVOC documentation.
- If you configure the address of the WebSocket tunnel server (see the 'Address' parameter below) as a domain name, you also need to configure the address of the DNS server that you want to use for resolving the domain name into an IP address. This is configured in the IP Interfaces table for the corresponding IP Interface (**OAMP**), using the 'Primary DNS' parameter and optionally, the 'Secondary DNS' parameter (see Configuring IP Network Interfaces on page 124).
- When the device is configured for WebSocket tunneling with OVOC, the SBC Configuration Wizard (see SBC Configuration Wizard on page 927) is not supported (and not accessible from the Web interface).

The following procedure describes how to configure WebSocket tunneling on the device through the Web interface. You can also configure it through CLI (`configure network > ovoc-tunnel-settings`).

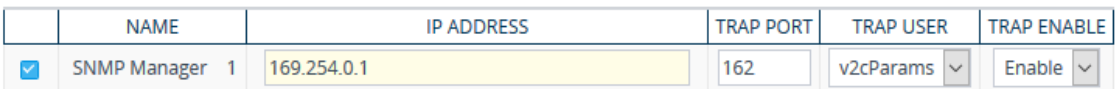
➢ **To configure WebSocket tunneling with OVOC on the device:**

1. Obtain the OVOC server's default certificate (trusted root certificate) for Managed Devices, and then import (see Importing Certificates into Trusted Root Certificate Store on page 173) the certificate into the device's Trusted Root store of the default TLS Context (ID #0).
2. Open the Web Service Settings page (**Setup** menu > **IP Network** tab > **Web Services** folder > **Web Service Settings**), and then under the OVOC Tunnel group, configure the following parameters:



- 'OVOC WebSocket Tunnel Server Address' [WSTunServer]: Configure it to the IP address or hostname of the OVOC server. If you configure the parameter to a hostname, the device uses the DNS server configured in Configuring a DNS Server for HTTP Services on page 343 to resolve it into an IP address.
- 'Path' [WSTunServerPath]: Configure it to "tun" (without quotation marks) to match the default OVOC configuration.
- 'Username' [WSTunUsername]: Configure it to match the WebSocket Tunnel username configured on OVOC. The default username is "VPN" (without quotation marks).
- 'Password' [WSTunPassword]: Configure it to match the WebSocket Tunnel password configured on OVOC. The default password is "123456" (without quotation marks).
- 'Secured (HTTPS)' [WSTunSecured]: Enable the parameter to use secure (HTTPS) transport for the WebSocket tunnel connection.
- 'Verify Certificate' [WSTunVerifyPeer]: Enable the parameter so that the device verifies the TLS certificate presented by OVOC during the establishment of the WebSocket tunnel connection.

3. Open the SNMP Trap Destinations table (see Configuring SNMP Trap Destinations with IP Addresses on page 80), and then configure an SNMP trap manager with IP address 169.254.0.1.



IP address 169.254.0.1 represents the OVOC server in the WebSocket tunnel overlay network.

4. For sending Quality of Experience (QoE) voice metric reports to OVOC, open the Quality of Experience Settings table (see Configuring OVOC for Quality of Experience on page 390), and then configure the 'OVOC Address' parameter to IP address 169.254.0.1.

To configure WebSocket tunneling on OVOC, refer to One Voice Operations Center IOM Manual.

9 INI File-Based Management

You can configure the device through an ini file, which is a text-based file with an *.ini file extension name, created using any standard text-based editor such as Notepad. Once you have created an ini file with all your configuration settings, you need to install (load) it to the device to apply the configuration. For a list of the *ini* file parameters, see Configuration Parameters Reference.

INI File Format

There are two types of *ini* file parameters:

- Individual parameters - see Configuring Individual ini File Parameters
- Table parameters - see Configuring Table ini File Parameters

Configuring Individual ini File Parameters

The syntax for configuring individual *ini* file parameters in the ini file is as follows:

- An optional, subsection name (or group name) enclosed in square brackets "[...]". This is used to conveniently group similar parameters by their functionality.
- Parameter name, followed by an equal "=" sign and then its value.
- Comments must be preceded by a semicolon ";".

```
[optional subsection name]
parameter name = value
parameter name = value
; this is a comment line
```

```
; for example:
[System Parameters]
SyslogServerIP = 10.13.2.69
EnableSyslog = 1
```

For general *ini* file formatting rules, see General ini File Formatting Rules.

Configuring Table ini File Parameters

Table ini file parameters allow you to configure tables, which include multiple parameters (*columns*) and row entries (*indices*). The table ini file parameter is composed of the following elements:

- **Table title:** The name of the table in square brackets, e.g., [MY_TABLE_NAME].

- **Format line:** Specifies the columns of the table (by their string names) that are to be configured.
  - The first word of the Format line must be "FORMAT", followed by the Index field name and then an equal "=" sign. After the equal sign, the names of the columns are listed.
  - Columns must be separated by a comma ",".
  - The Format line must only include columns that can be modified (i.e., parameters that are not specified as read-only). An exception is Index fields, which are mandatory.
  - The Format line must end with a semicolon ";".
- **Data line(s):** Contain the actual values of the columns (parameters). The values are interpreted according to the Format line.
  - The first word of the Data line must be the table's string name followed by the Index field.
  - Columns must be separated by a comma ",".
  - A Data line must end with a semicolon ";".
- **End-of-Table Mark:** Indicates the end of the table. The same string used for the table's title, preceded by a backslash "\", e.g., [\MY_TABLE_NAME].

The following displays an example of the structure of a table ini file parameter:

```
[Table_Title]
; This is the title of the table.
FORMAT Index = Column_Name1, Column_Name2, Column_Name3;
; This is the Format line.
Index 0 = value1, value2, value3;
Index 1 = value1, value2, value3;
; These are the Data lines.
[\Table_Title]
; This is the end-of-the-table-mark.
```

- The table ini file parameter formatting rules are listed below:
- Indices (in both the Format and the Data lines) must appear in the same order. The Index field must never be omitted.
- The Format line can include a subset of the configurable fields in a table. In this case, all other fields are assigned with the pre-defined default values for each configured line.
- The order of the fields in the Format line isn't significant (as opposed to the Index fields). The fields in the Data lines are interpreted according to the order specified in the Format line.
- The order of the Data lines is insignificant.

- Data lines must match the Format line, i.e., it must contain exactly the same number of Indices and Data fields and must be in exactly the same order.
- A row in a table is identified by its table name and Index field. Each such row may appear only once in the *ini* file.
- Table dependencies: Certain tables may depend on other tables. For example, one table may include a field that specifies an entry in another table. This method is used to specify additional attributes of an entity, or to specify that a given entity is part of a larger entity. The tables must appear in the order of their dependency (i.e., if Table X is referred to by Table Y, Table X must appear in the *ini* file before Table Y).

The table below displays an example of a table ini file parameter:

```
[ CodersGroup0 ]
FORMAT CodersGroup0_Index = CodersGroup0_Name, CodersGroup0_pTime,
CodersGroup0_rate, CodersGroup0_PayloadType, CodersGroup0_Sce,
CodersGroup0_CoderSpecific;
CodersGroup0 0 = g711Alaw64k, 20, 0, 255, 0, 0;
CodersGroup0 1 = eg711Ulaw, 10, 0, 71, 0, 0;
[ \CodersGroup0 ]
```

Don't include read-only parameters in table ini file parameters. This can cause an error when loading the file to the device.

General ini File Formatting Rules

The *ini* file must adhere to the following formatting rules:

- The *ini* file name must not include hyphens "-" or spaces; if necessary, use an underscore "_ " instead.
- Lines beginning with a semi-colon ";" are ignored. These can be used for adding remarks in the *ini* file.
- A carriage return (i.e., Enter) must be done at the end of each line.
- The number of spaces before and after the equals sign "=" is irrelevant.
- Subsection names for grouping parameters are optional.
- If there is a syntax error in the parameter name, the value is ignored.
- Syntax errors in the parameter's value can cause unexpected errors (parameters may be set to the incorrect values).
- Parameter string values that denote file names (e.g., CallProgressTonesFileName) must be enclosed with inverted commas, e.g., CallProgressTonesFileName = 'cpt_usa.dat'.
- The parameter name is not case-sensitive.

- The parameter value is not case-sensitive, except for coder names.
- The *ini* file must end with at least one carriage return.

Configuring an ini File

There are different methods that you can use for configuring an ini file before you load it to the device.

- Modifying the device's current ini file: This method is recommended if you mainly need to change the settings of parameters that you have previously configured.
  a. Save the device's current configuration as an *ini* file on your computer, using the Web interface (see Saving Configuration).
  b. Open the file using a text file editor, and then modify the *ini* file as required.
  c. Save and close the file.
  d. Load the file to the device.
- Creating a new ini file that includes only updated configuration:
  a. Open a text file editor such as Notepad.
  b. Add only the required parameters and their settings.
  c. Save the file with the ini file extension name (e.g., myconfiguration.ini).
  d. Load the file to the device.

For loading ini files to the device, see Loading an ini File to the Device.

- If you save an ini file from the device and a table row is configured with invalid values, the ini file displays the row prefixed with an exclamation mark (!), for example:

  !CpMediaRealm 1 = "ITSP", "Voice", "", 60210, 2, 6030, 0, "", "";

- To restore the device to default settings through the *ini* file, see Restoring Factory Defaults.

Loading an ini File to the Device

You can load an *ini* file to the device using the following methods:

- CLI:
  - Configuration:
    - To apply the parameter settings of the file and restore parameters that are not included in the file to default settings:

```
# copy ini-file from <URL>
```

- To apply the parameter settings of the file and keep the current settings of parameters that are not included in the file:

```
# copy incremental-ini-file from <URL>
```

- Web interface:
  - Auxiliary Files page (see Loading Auxiliary Files): The device updates its configuration according to the loaded ini file while preserving the remaining current configuration.
  - Configuration File page (see Configuration File): The device updates its configuration according to the loaded ini file and applies default values to parameters that were not included in the loaded ini file.

When you load an ini file to the device, its configuration settings are saved to the device's non-volatile memory (flash).

> Before you load an *ini* file to the device, make sure that the file extension name is **.ini*.

Secured Encoded ini File

The *ini* file contains sensitive information that is required for the functioning of the device. The file may be loaded to the device using HTTP. These protocols are not secure and are vulnerable to potential hackers. To overcome this security threat, the AudioCodes DConvert utility allows you to binary-encode (encrypt) the *ini* file before loading it to the device. For more information, refer to the *DConvert Utility User's Guide*.

> If you save an ini file from the device to a folder on your PC, an *ini* file that was loaded to the device encoded is saved as a regular *ini* file (i.e., unencoded).

Configuring Password Display in ini File

Passwords can be displayed in the ini file (saved from the Web interface or CLI) as obscured (encrypted) or hidden:

- Obscured: (Default) The password characters are concealed and displayed as encoded. The password is displayed using the syntax, *1<obscured password>*, for example, 1S3p+fno=.
- Hidden: the password is replaced with an asterisk (*).

To configure the desired format, use the [INIPasswordsDisplayType] parameter.

- When you load an ini file to the device containing obscured passwords, the passwords are parsed and applied to the device.
- When you load an ini file to the device containing hidden passwords, the passwords are ignored.
- By default, the format is obscured passwords, thus enabling their full recovery in case of configuration restore or copy to another device.
- Regardless of the configured password display format, the View mode in the INI Viewer & Editor utility also displays the passwords in plain text (in parenthesis), as shown in the below example:

```
[SNMP Params]

SnmpReadWriteCommunityStringsPassword_0 = $1$+JKWkpXNz83L (john1234)
```

INI Viewer and Editor Utility

For more information on AudioCodes INI Viewer and Editor utility, refer to the *INI Viewer & Editor User's Guide*.

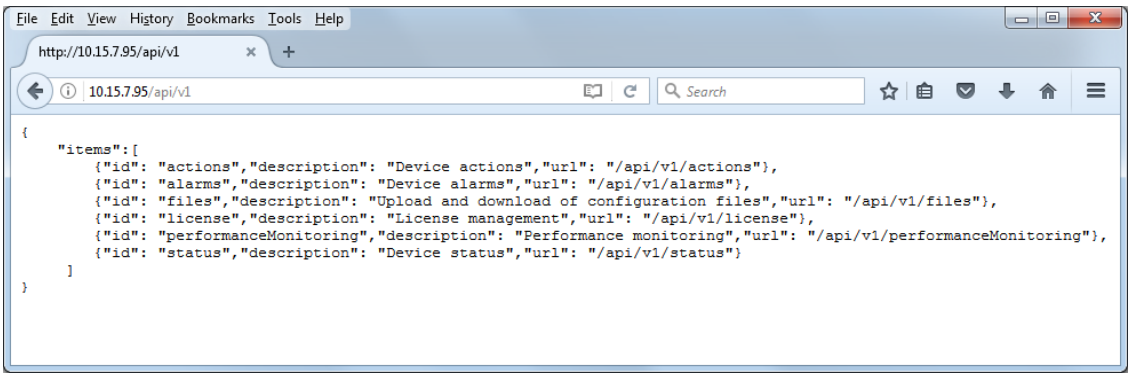
10 REST-Based Management

You can manage the device through the Representational State Transfer (REST) architecture. REST is a Web-based access service, allowing you to access the device's management interface over HTTP/S. Developers can use the device's REST API to integrate the device into their solution and allow administrators to perform management and configuration tasks through automation scripts. The REST API also displays performance monitoring counters.

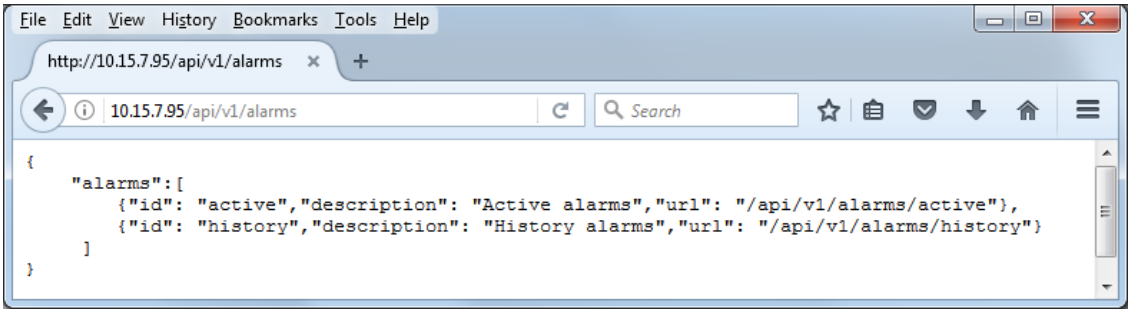
The REST API relies on a simple pre-defined URL path (**<device's OAMP IP address>/api/v1**) through which device resources can be accessed. Each resource represents a specific device management element (e.g., file upload), state object (e.g., alarms), or maintenance action (e.g., reset). The REST API uses the standard HTTP/1.1 protocol. Standard HTTP methods (GET, PUT, POST and DELETE) are used to read the resource's state and to create, update, and delete the resources, respectively. Resource state is described in JSON format and included in the HTTP request or response bodies. For security, it is recommended to secure REST traffic by using HTTPS (see the [HTTPSOnly] parameter).

➢ To access the REST API:

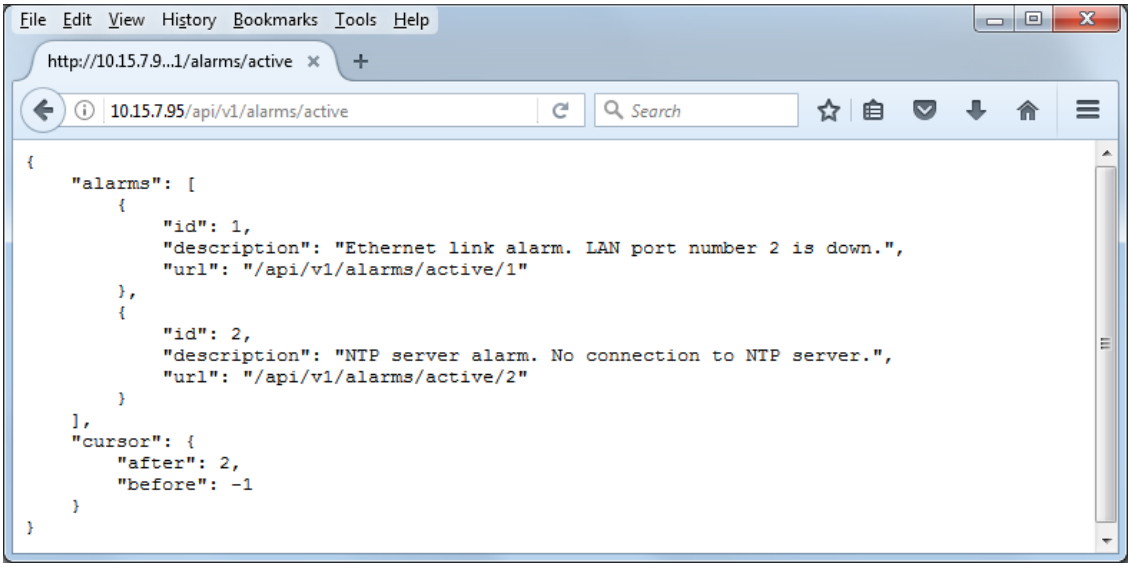
1. Open a standard Web browser, and then in the URL field, enter the device's OAMP IP address followed by "/api/v1" (e.g., 10.15.7.95**/api/v1**); you are prompted to enter your login credentials.
2. Enter your login username and password, and then click **Sign In**; the device's REST interface appears, showing the URL paths of the different resource items:



3. Access the required resource item using the shown URL. For example, to access the device's alarms resource, append "/alarms" to the URL (i.e. 10.15.7.95/api/v1**/alarms**). Some items have sub-resources such as the alarms item. When you access the alarms item, the URLs to the active and history alarms resources are shown.



4. To access a sub-resource (e.g., active alarms) if exists, use the shown URL. For example, to access the active alarms resource, append "/active" to the URL (i.e. 10.15.7.95/api/v1/alarms**/active**).



- If you know the URL of the resource, instead of accessing each resource menu, you can access it directly using the full URL path (e.g., **/api/v1/alarms/active**).
- For more information on REST API, refer to the document *REST API for Mediant Devices*.
- When accessing the device's REST interface, you are prompted for your management user credentials (username and password).

Part III

General System Settings

11 Date and Time

The device's internal clock (date and time) can be set using one of the following methods:

- Automatically synchronized using a third-party, remote Simple Network Time Protocol (SNTP) server (see Configuring Automatic Date and Time through SNTP below)
- Manually (see Configuring Manual Date and Time on page 106)

Configuring Automatic Date and Time through SNTP

The device's Simple Network Time Protocol (SNTP) client functionality generates requests and reacts to the resulting responses using the NTP Version 3 protocol definitions (according to RFC 1305). Through these requests and responses, the device, as an NTP client, synchronizes the system time to a time source within the network, thereby eliminating any potential issues should the local system clock "drift" during operation. The NTP client follows a simple process in managing system time: 1) the NTP client requests an NTP update, 2) receives an NTP response and then 3) updates the local system clock based on an NTP server within the network. The client requests a time update from the user-defined NTP server (IP address or FQDN) at a user-defined update interval. Typically, the update interval is every 24 hours based on when the system was restarted.

You can also configure the device to authenticate and validate NTP messages received from the NTP server. Authentication is done using an authentication key with the MD5 cryptographic hash algorithm. When this feature is enabled, the device ignores NTP messages received without authentication.

The following procedure describes how to configure SNTP through the Web interface. For detailed descriptions of the configuration parameters, see NTP and Daylight Saving Time Parameters.

➢ **To configure SNTP through the Web interface:**

1. Open the Time & Date page (**Setup** menu > **Administration** tab > **Time & Date**), and then scroll down to the NTP Server group:

2. Configure the NTP server address:

- In the 'Primary NTP Server Address' [NTPServerIP] field, configure the primary NTP server's address (IP or FQDN).
- (Optional) In the 'Secondary NTP Server Address' [NTPSecondaryServerIP] field, configure the backup NTP server.

3. In the 'NTP Updated Interval' [NTPUpdateInterval] field, configure the period after which the date and time of the device is updated.
4. Configure NTP message authentication:
   - In the 'NTP Authentication Key Identifier' field, configure the NTP authentication key identifier.
   - In the 'NTP Authentication Secret Key' field, configure the secret authentication key shared between the device and the NTP server.
5. Click **Apply**.
6. Verify that the device has received the correct date and time from the NTP server. The date and time is displayed in the 'UTC Time' read-only field under the Time Zone group.

If the device does not receive a response from the NTP server, it polls the NTP server for 10 minutes. If there is still no response after this duration, the device declares the NTP server as unavailable and raises an SNMP alarm [acNTPServerStatusAlarm]. The failed response could be due to incorrect configuration.

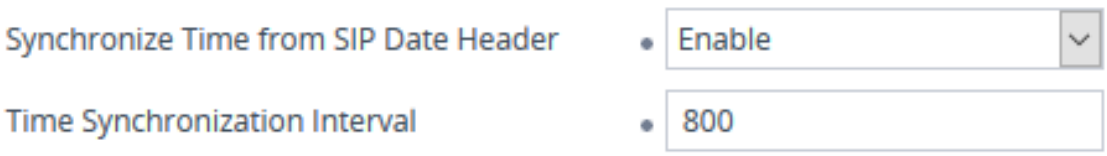
Configuring Automatic Date and Time through SIP

You can configure the device to synchronize its internal clock (date and time) with a remote SIP endpoint (according to RFC 3261). When enabled, the device obtains the date and time from the Date header in the incoming 200 OK message received in response to a REGISTER request sent by the device. This can be any REGISTER request sent for normal SIP traffic handling (i.e., it's not a specific REGISTER message that is sent to a specific SIP server or endpoint). An example of a SIP Date header with date and time is shown below:

```
Date: Sat, 12 Mar 2020 23:29:00 GMT
```

➢ **To configure clock synchronization through SIP:**

1. Open the Time & Date page (**Setup** menu > **Administration** tab > **Time & Date**), and then scroll down to the Date Header Time Sync group:



2. In the 'Synchronize Time from SIP Date Header' [DateHeaderTimeSync] field, select **Enable** to enable the feature.
3. In the 'Time Synchronization Interval' [DateHeaderTimeSyncInterval] field, enter the minimum time (in seconds) between synchronization updates. For example, if configured to 8640 (24 hours) and the device receives within this 24-hour interval a SIP response to a REGISTER with the Date header, it ignores the date. Only if it receives such a header after this interval does it update its clock according to the header, and then does the next update 24 hours later.
4. Click **Apply**. When the device receives a SIP response with the Date header, it updates its clock and the date and time is displayed in the 'UTC Time' read-only field under the Time Zone group.

- The device only uses the date and time in the SIP Date header if its value is year 2016 or later.
- If you have enabled clock synchronization using an NTP server (see Configuring Automatic Date and Time through SNTP on page 104) and using the SIP Date header, synchronization using the NTP server takes precedence (i.e., device ignores received Date headers). When both are enabled, the device sends the SNMP alarm acClockConfigurationAlarm.
- Once a week, the device stores the clock's date and time in its flash memory. If the device is restarted, its clock is set to this stored date and time, and updated once it receives a Date header in a SIP response to a sent REGISTER message.

Configuring Manual Date and Time

You can manually configure the date and time of the device instead of using an NTP server (as described in Configuring Automatic Date and Time using SNTP).

➢ **To manually configure the device's date and time through the Web interface:**

1. Open the Time & Date page (**Setup** menu > **Administration** tab > **Time & Date**), and then scroll down to the Local Time group:

2. Configure the current date and time of the geographical location in which the device is installed:
   - Date:
     - 'Year' in *yyyy* format (e.g., "2015")
     - 'Month' in *mm* format (e.g., "3" for March)
     - 'Day' in *dd* format (e.g., "27")

- Time:
  - 'Hours' in 24-hour format (e.g., "4" for 4 am)
  - 'Minutes' in *mm* format (e.g., "57")
  - 'Seconds' in *ss* format (e.g., "45")

3. Click **Apply**; the date and time is displayed in the 'UTC Time' read-only field.

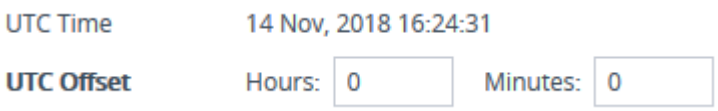
- If the device is configured to obtain date and time from an NTP server, the fields under the Local Time group are read-only, displaying the date and time received from the NTP server.
- After performing a hardware reset, the date and time are returned to default values and thus, you should subsequently update the date and time.

Configuring the Time Zone

You can configure the time zone in which the device is deployed. This is referred to as the Coordinated Universal Time (UTC) time offset and defines how many hours the device is from Greenwich Mean Time (GMT). For example, Germany Berlin is one hour ahead of GMT (UTC/GMT is +1 hour) and therefore, you would configure the offset to "1". USA New York is five hours behind GMT (UTC/GMT offset is -5 hours) and therefore, you would configure the offset as a minus value "-5".

➢ **To configure the time zone:**

1. Open the Time & Date page (**Setup** menu > **Administration** tab > **Time & Date**), and then scroll down to the Time Zone group:



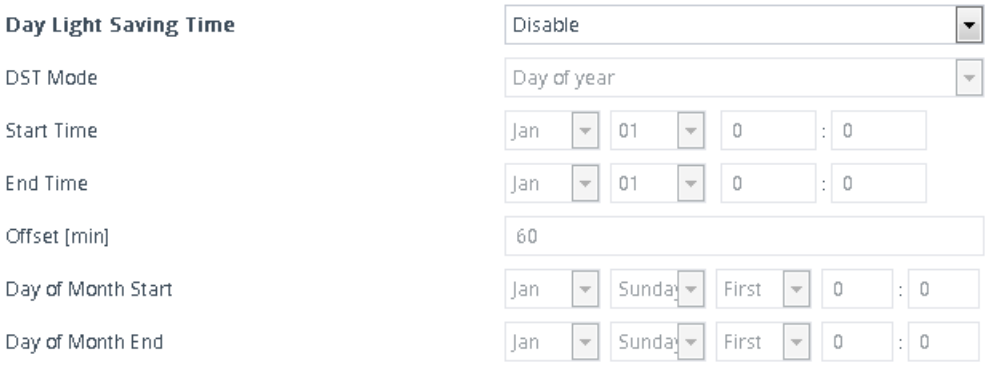
2. In the 'UTC Offset' fields (NTPServerUTCOffset), configure the time offset in relation to the UTC. For example, if your region is GMT +1 (an hour ahead), enter "1" in the 'Hours' field.
3. Click **Apply;** the updated time is displayed in the 'UTC Time' read-only field and the fields under the Local Time group.

Configuring Daylight Saving Time

You can apply daylight saving time (DST) to the date and time of the device. DST defines a date range in the year (summer) where the time is brought forward so that people can experience more daylight. DST applies an offset of up to 60 minutes (default) to the local time. For example, Germany Berlin has DST from 30 March to 26 October, where the time is brought forward by an hour (e.g., 02:00 to 03:00 on 30 March). Therefore, you would configure the DST offset to 60 minutes (one hour).

➢ To configure DST through the Web interface:

1. Open the Time & Date page (**Setup** menu > **Administration** tab > **Time & Date**), and then scroll down to the Time Zone group:



2. From the 'Day Light Saving Time' (DayLightSavingTimeEnable) drop-down list, select **Enable**.
3. From the 'DST Mode' drop-down list, select the range type for configuring the start and end dates for DST:
   - **Day of year:** The range is configured by exact date (day number of month), for example, from March 30 to October 30. If 'DST Mode' is set to **Day of year**, in the 'Start Time' (DayLightSavingTimeStart) and 'End Time' (DayLightSavingTimeEnd) drop-down lists, configure the period for which DST is relevant.
   - **Day of month:** The range is configured by month and day type, for example, from the last Sunday of March to the last Sunday of October. If 'DST Mode' is set to **Day of month**, in the 'Day of Month Start' and 'Day of Month End' drop-down lists, configure the period for which DST is relevant.
4. In the 'Offset' (DayLightSavingTimeOffset) field, configure the DST offset in minutes.
5. If the current date falls within the DST period, verify that it has been successful applied to the device's current date and time. You can view the device's date and time in the 'UTC Time' read-only field.

12 Configuring a Hostname for the Device

You can configure a hostname (FQDN) for the device. This hostname should also be defined at a DNS server so that when queried, the DNS can resolve the hostname into the device's correct IP address.

A configured hostname affects the following:

- The device's Web interface and CLI (remotely using Telnet/SSH) can be accessed (logged in) using the hostname (instead of the OAMP IP address). For example, when logging into the Web interface through HTTP, you would enter the hostname in your Web browser like this: http://<hostname>.
  - Web: The toolbar displays the hostname (first 16 characters only) instead of the device type
  - CLI: The CLI prompt displays the hostname instead of the device type.
- The device's SNMP interface's SysName object (under MIB-2) is set to the hostname.
- TLS certificates used by the device for HTTPS-based communication with AudioCodes OVOC are issued with a hostname (instead of an IP address). For certificate signing requests (CSR) with a Certification Authority (CA), the hostname is used as the Common Name (CN or Subject Name) and Subject Alternative Name (SAN). For configuring CSRs, see Assigning CSR-based Certificates to TLS Contexts on page 164.
- In HA systems, the device-pair share the same hostname.

➢ To configure a hostname for the device:

1. Open the Network Settings page (**Setup** menu > **IP Network** tab > **Advanced** folder > **Network Settings**).
2. In the 'Host Name' field [Hostname], enter the hostname.

3. Click **Apply**.

> If you configure the device with a hostname, you can protect the device against DNS rebinding attacks whenever you access the device with the hostname. For more information, see Enabling DNS Rebinding Protection on page 66.

Part IV

General VoIP Configuration

13 Network

This section describes network-related configuration.

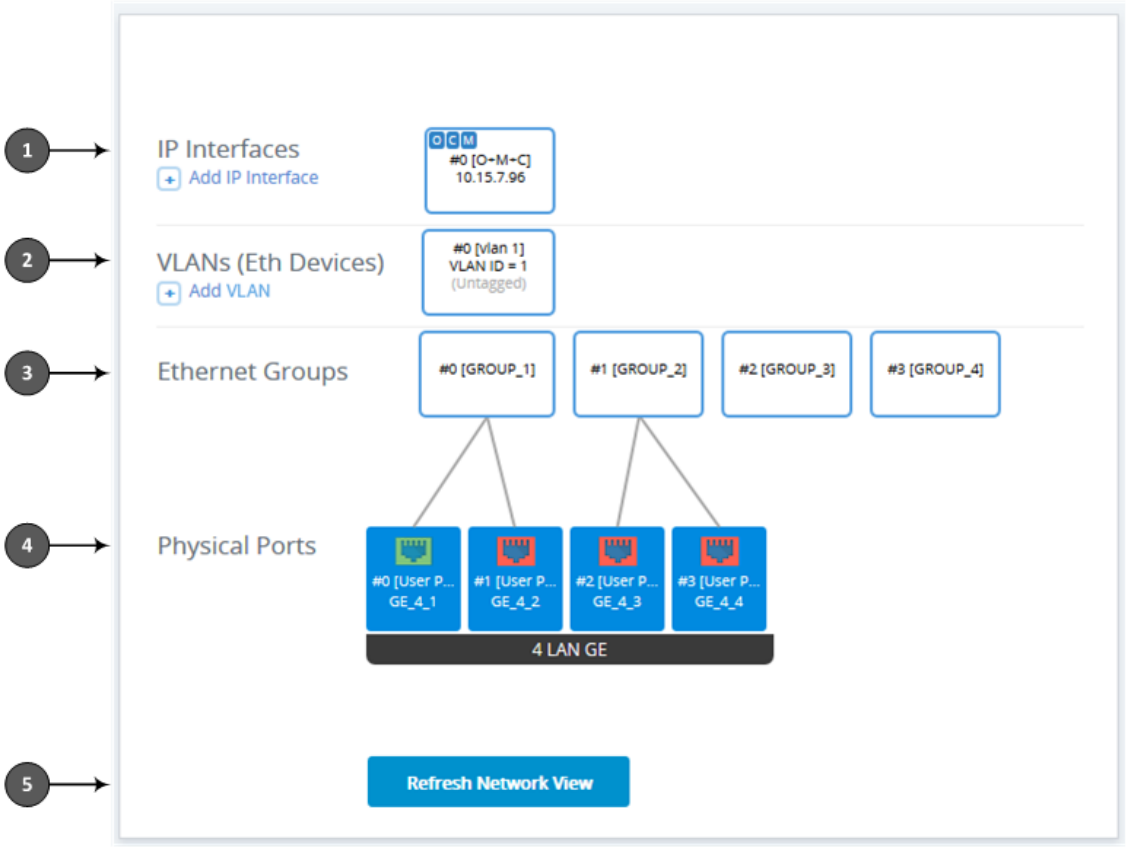
Building and Viewing your Network Topology

The Network view lets you easily build and view your voice network topology entities, including IP network interfaces, Ethernet Devices (VLANs), Ethernet Groups, and physical Ethernet ports. The Network view graphically displays these entities and the associations between them, giving you a better understanding of your network topology and configuration. You can use the Network view as an alternative to configuring the entities in their respective Web pages or you can use it in combination.

➢ **To access the Network view:**

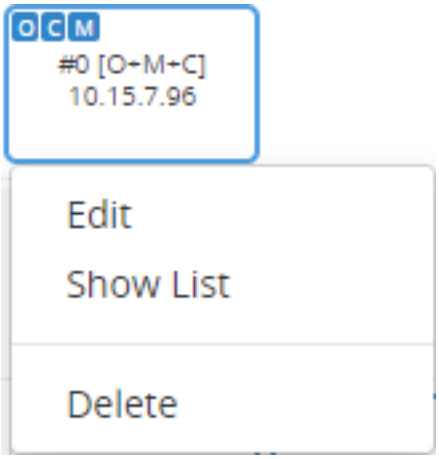
- Click the Network View home icon (**Setup** menu > **IP Network** tab > **Network View**).

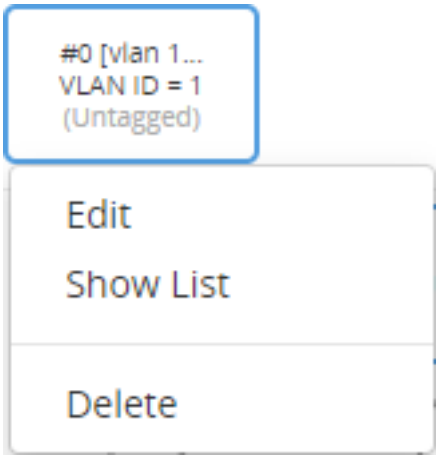
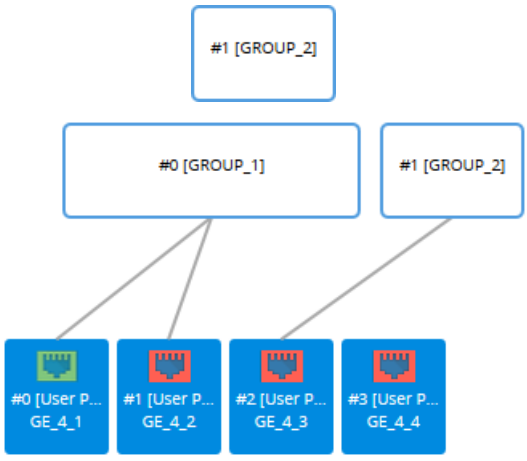
The areas of the Network view is shown in the example below and described in the subsequent table.



The figure above is used only as an example; your device may show different Ethernet Groups and Ethernet ports.

Table 13-1: Description of Network View

Item	Description
1	Displays configured IP Interfaces (see Configuring IP Network Interfaces). The IP Interface appears as an icon, displaying the application type ("OCM" for OAMP, "C" for Control, and "M" for Media), row index number, name, and IP address. If you click the icon, a drop-down menu appears, listing commands:  ■ **Edit:** Opens a dialog box in the IP Interfaces table to modify the IP Interface. ■ **Show List:** Opens the IP Interfaces table, allowing you to configure IP Interfaces. ■ **Delete:** Opens the IP Interfaces table where you are prompted to confirm deletion of the IP Interface. **To add an IP Interface:** 1. Click **Add IP Interface**; the IP Interfaces table opens with a new dialog box for adding an IP Interface to the next available index row. 2. Configure the IP Interface as desired, and then click **Apply**; the IP Interfaces table closes and you are returned to the Network View, displaying the newly added IP Interface.
2	Displays configured Ethernet Devices (see Configuring Underlying Ethernet Devices). The Ethernet Device appears as an icon, displaying the row index number, name, VLAN ID and if its tagged or untagged. If you click the icon, a drop-down menu appears, listing commands:

Item	Description
	 ■ **Edit:** Opens a dialog box in the Ethernet Devices table to modify the Ethernet Device. ■ **Show List:** Opens the Ethernet Devices table, allowing you to configure all Ethernet Devices. ■ **Delete:** Opens the Ethernet Devices table where you are prompted to confirm deletion of the Ethernet Device. **To add an Ethernet Device:** 1. Click **Add VLAN**; the Ethernet Devices table opens with a new dialog box for adding an Ethernet Device to the next available index row. 2. Configure the Ethernet Devices as desired, and then click **Apply**; the Ethernet Devices table closes and you are returned to the Network View, displaying the newly added Ethernet Device.
3	Displays configured Ethernet Groups (see Configuring Ethernet Port Groups). The Ethernet Groups appear as icons, displaying the row index number, and name. Ethernet ports associated with Ethernet Groups are indicated by lines connecting between them, as shown in the example below: 

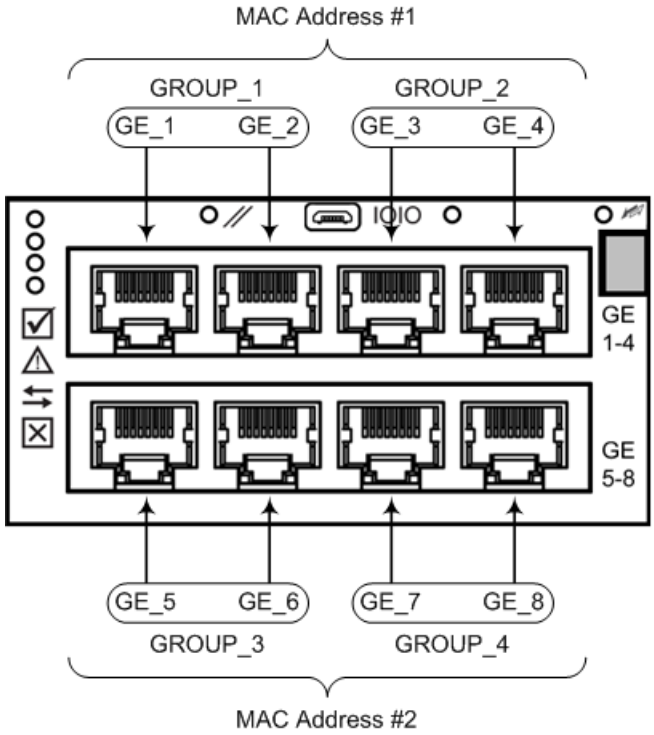
Item	Description
	To edit an Ethernet Group: 1. Click the Ethernet Group icon, and then from the drop-down menu, choose **Edit**; the Ethernet Groups table opens with a dialog box for editing the Ethernet Group. 2. Configure the Ethernet Group as desired, and then click **Apply**; the Ethernet Groups table closes and you are returned to the Network View. To open the Ethernet Groups table, click any Ethernet Group icon, and then from the drop-down menu, choose **Show List**. You can then view and edit all the Ethernet Groups in the table.
4	Configures and displays the device's Ethernet ports. **To configure an Ethernet port:** 1. Click the required port icon, and then from the drop-down menu, choose **Edit**; the Physical Ports table opens with a dialog box for editing the Ethernet port. 2. Configure the Ethernet Port as desired, and then click **Apply**; the Physical Ports table closes and you are returned to the Network View. For more information on configuring Ethernet ports, see Configuring Underlying Ethernet Devices. The Ethernet ports appear as icons, displaying the row index number, description, and port string number, as shown in the example below:  The connectivity status of the port is indicated by the color of the icon: ■ Green: Network connectivity exists through port (port connected to network). ■ Red: No network connectivity through port (e.g., cable disconnected). To refresh the status indication, click the **Refresh Network View** button (described below in Item #5). To open the Physical Ports table, click any port icon, and then from the drop-down menu, choose **View List**. You can then view and edit all the ports in the table.
5	If you keep the Network view page open for a long time, you may want to click the **Refresh Network View** button to refresh the connectivity status display of the Ethernet ports.

Configuring Physical Ethernet Ports

The Physical Ports table lets you configure the device's Ethernet ports. This includes configuring port speed and duplex mode (half or full), and a brief description of the port. The table also displays the status of the port as well as the port group (*Ethernet Group*) to which the port belongs. For more information on Ethernet Groups, see Configuring Ethernet Port Groups.

The names of the ports displayed in the device's management tools (e.g., Web interface) are different to the labels of the physical ports on the chassis. The figure below shows the mapping between the two:

Figure 13-1: Mediant 9000Ethernet Port String Names



You can also view the mapping of the ports, using the following CLI command:

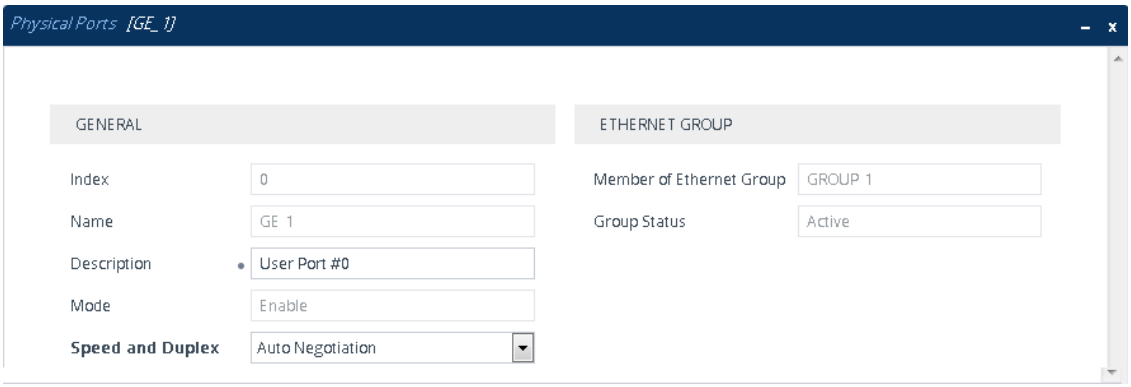
```
# show network physical-port
```

- The device provides two MAC addresses for the LAN ports which are allocated as follows: a MAC address for ports GE_1 through GE_4, and a MAC address for ports GE_5 through GE_8.
- If you are connecting to the same switch, ports with the same MAC address (e.g., GE_1 and GE_3) and belonging to different Ethernet Groups, each of these Ethernet Groups must have a unique tagged VLAN ID.

The following procedure describes how to configure Ethernet ports through the Web interface. You can also configure it through ini file [PhysicalPortsTable] or CLI (`configure network > physical-port`).

➢ To configure the physical Ethernet ports:

1. Open the Physical Ports table (**Setup** menu > **IP Network** tab > **Core Entities** folder > **Physical Ports**).
2. Select a port that you want to configure, and then click **Edit**; the following dialog box appears:



3. Configure the port according to the parameters described in the table below.
4. Click **Apply**, and then reset the device with a burn-to-flash for your settings to take effect.

Table 13-2: Physical Ports Table Parameter Descriptions

Parameter	Description
General	
'Index'	(Read-only) Displays the index number for the table row.
'Name' `port` [PhysicalPortsTable_Port]	(Read-only) Displays the Ethernet port number. See the figure in the beginning of this section for the mapping between the GUI port number and the physical port on the chassis.
'Description' `port-description` [PhysicalPortsTable_PortDescription]	Defines a description of the port. By default, the value is "User Port #<row index>". **Note:** Each row must be configured with a unique name.
'Mode' `mode` [PhysicalPortsTable_Mode]	(Read-only) Displays the mode of the port. ■ [0] **Disable** ■ [1] **Enable** (default)
'Speed and Duplex'	Defines the speed and duplex mode of the port.

Parameter	Description
`speed-duplex` [PhysicalPortsTable_SpeedDuplex]	■ [0] **10BaseT Half Duplex** ■ [1] **10BaseT Full Duplex** ■ [2] **100BaseT Half Duplex** ■ [3] **100BaseT Full Duplex** ■ [4] **Auto Negotiation** (default) ■ [6] **1000BaseT Half Duplex** ■ [7] **1000BaseT Full Duplex**
Ethernet Group	
'Member of Ethernet Group' `group-member` [PhysicalPortsTable_GroupMember]	(Read-only) Displays the Ethernet Group to which the port belongs. To assign the port to a different Ethernet Group, see Configuring Ethernet Port Groups.
'Group Status' `group-status` [PhysicalPortsTable_GroupStatus]	(Read-only) Displays the status of the port: ■ "Active": Active port. When the Ethernet Group includes two ports and their transmit/receive mode is configured to **2RX 1TX** or **2RX 2TX**, both ports show "Active". ■ "Redundant": Standby (redundant) port.

Configuring Ethernet Port Groups

The Ethernet Groups table lets you configure Ethernet Groups. An Ethernet Group represents a physical Ethernet port(s) on the device. You can assign an Ethernet Group with one, two, or no ports (*members*). When two ports are assigned to an Ethernet Group, 1+1 Ethernet port redundancy can be implemented in your network.In such a configuration, one port can be active while the other standby or both ports can be active, depending on the ports' transmit (Tx) and receive (Rx) settings. This provides port redundancy within the Ethernet Group, whereby if a port is disconnected the device switches over to the other port in the Ethernet Group. If you configure an Ethernet Group with only one port, the Ethernet Group operates as a single port (no redundancy).

The Ethernet Groups table also lets you configure the transmit (Tx) and receive (Rx) settings of the Ethernet ports per Ethernet Group. The Tx/Rx setting is applicable only to Ethernet Groups that contain two ports. This setting determines if both ports or only one of the ports can receive and transmit traffic.

The maximum number of Ethernet Groups that you can configure is the same as the number of Ethernet ports provided by the device. Thus, the device supports up to eight Ethernet Groups. You can assign one or two ports to an Ethernet Group. By default, each Ethernet Group is assigned twoports (Ethernet Group 2 which is assigned only port 0/3); the other Ethernet Groups are empty. For default port assignment to Ethernet Groups, see Configuring Physical Ethernet Ports.

You can assign Ethernet ports to IP network interfaces. This is done by first configuring an Ethernet Device with the required Ethernet Group containing the port or ports (see Configuring Underlying Ethernet Devices). Then by assigning the Ethernet Device to the IP network interface in the IP Interfaces table (see Configuring IP Network Interfaces). This enables physical separation of network interfaces, providing a higher level of segregation of sub-networks. Equipment connected to different physical ports is not accessible to one another; the only connection between them can be established by cross connecting them with media streams (VoIP calls).

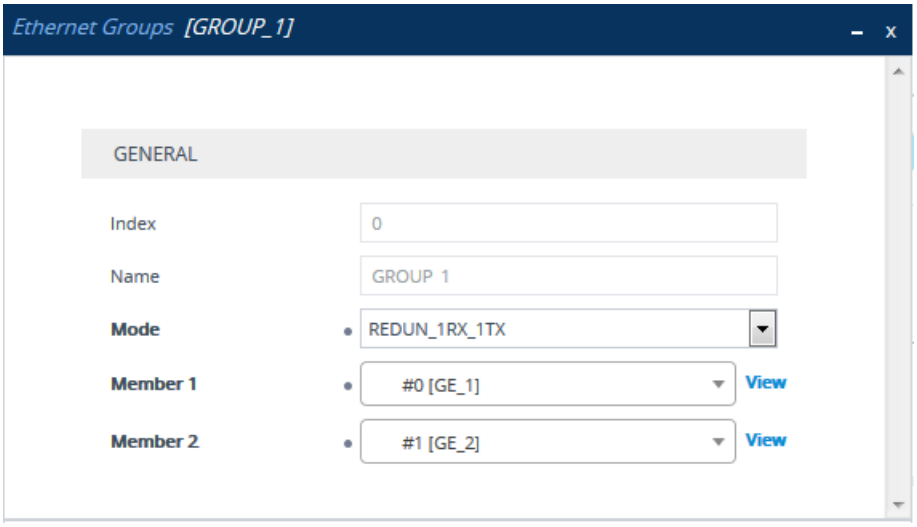
The port names (strings) displayed in the Ethernet Groups table represent the physical ports on the device. For the mapping of these strings to the physical ports, see Configuring Physical Ethernet Ports.

- If you want to assign a port to a different Ethernet Group, you must first remove the port from its current Ethernet Group. To remove the port, configure the 'Member' field so that no port is selected or select a different port.
- Two different MAC addresses are assigned to the Ethernet ports: one to ports GE 1-4 (upper ports) and another to ports GE 5-8 (lower ports).
- If you configure an Ethernet Group with two port members, the ports must belong to the same MAC address (see note above - – both GE 1-4 or both GE 5-8). For example, you can configure an Ethernet Group with ports 1 and 3, but not with ports 1 and 5.
- Ports with the same MAC address (e.g., GE 1-4 ports) must each be connected to a different Layer-2 switch.
- When implementing 1+1 Ethernet port redundancy, each port in the Ethernet Group (port pair) must be connected to a different switch (but in the same subnet).

The following procedure describes how to configure Ethernet Groups through the Web interface. You can also configure it through ini file [EtherGroupTable] or CLI (`configure network > ether-group`).

➢ **To configure Ethernet Groups:**

1. Open the Ethernet Groups table (**Setup** menu > **IP Network** tab > **Core Entities** folder > **Ethernet Groups**).
2. Select the Ethernet Group that you want to configure, and then click **Edit**; the following dialog box appears:



3. Configure the Ethernet Group according to the parameters described in the table below.
4. Click **Apply**, and then reset the device with a burn-to-flash for your settings to take effect.

Table 13-3: Ethernet Groups Table Parameter Descriptions

Parameter	Description
'Index'	(Read-only) Displays the index number for the table row.
'Name' `group` [EtherGroupTable_ Group]	(Read-only) Displays the Ethernet Group number.
'Mode' `mode` [EtherGroupTable_Mode]	Defines the mode of operation of the ports in the Ethernet Group. This applies only to Ethernet Groups containing two ports. ■ [0] **None** = Select this option to remove all ports from the Ethernet Group. ■ [1] **Single** = Select this option if the Ethernet Group contains only one port. ■ [2] **1RX/1TX** = (Default) At any given time, only one of the ports in the Ethernet Group transmits and receives packets. If a link exists on both ports, the active one is either the first to have a link up or the lower-numbered port if both have the same link up from start. ■ [3] **2RX/1TX** = Both ports in the Ethernet Group can receive packets, but only one port can transmit. The transmitting port is determined arbitrarily by the device. If the selected port fails at a later stage, a switchover to the

Parameter	Description
	redundant port is done, which begins to transmit and receive. ■ [4] **2RX/2TX** = Both ports in the Ethernet Group can receive and transmit packets. This option is applicable only to the Maintenance interface for High Availability (HA) deployments. For more information, see Network Topology Types and Rx/Tx Ethernet Port Group Settings. **Note:** ■ You can configure an Ethernet Group with the following port members only: ✓ GE_1 and GE_2 ✓ GE_3 and GE_4 ✓ GE_5 and GE_6 ✓ GE_7 and GE_8 ■ When implementing High Availability, it is recommended to use the **2RX/1TX** option for the Maintenance interface. For more information, see Initial HA Configuration.
'Member 1' `member1` [EtherGroupTable_ Member1]	Assigns the first port to the Ethernet Group. To assign no port, set this field to **None**. **Note:** Before you can re-assign a port to a different Ethernet Group, you must first remove the port from its current Ethernet Group. To remove the port, either set this field to **None** or to a different port.
'Member 2' `member2` [EtherGroupTable_ Member2]	Assigns the second port to the Ethernet Group. To assign no port, set this field to **None**. **Note:** Before you can re-assign a port to a different Ethernet Group, you must first remove the port from its current Ethernet Group. To remove the port, either set this field to **None** or to a different port.

Configuring Underlying Ethernet Devices

The Ethernet Devices table lets you configure up to 1,024 *Ethernet Devices*. An Ethernet Device represents a Layer-2 bridging device and is assigned a VLAN ID and an Ethernet Group (Ethernet port group). Multiple Ethernet Devices can be associated with the same Ethernet Group. The Ethernet Device (VLAN) can be configured with a VLAN tagging policy, which determines

whether the Ethernet Device accepts tagged or untagged packets received on the Ethernet port associated with the Ethernet Device.

Once configured, assign the Ethernet Device to an IP network interface in the IP Interfaces table ('Underlying Device' field) and/or with a static route in the Static Routes table ('Ethernet Output Device' field). You can assign the same Ethernet Device to multiple IP network interfaces and thereby, implement multi-homing (multiple addresses on the same interface/VLAN).

By default, the device provides a pre-configured Ethernet Device at Index 0 with the following settings:

- Name: "vlan 1"
- VLAN ID: 1
- Ethernet Group: GROUP 1
- Tagging Policy: Untagged
- MTU: 1500

The pre-configured Ethernet Device is associated with the default IP network interface (ie., OAMP) in the IP Interfaces table. The Untagged policy of the pre-configured Ethernet Device enables you to connect to the device using the default OAMP interface.

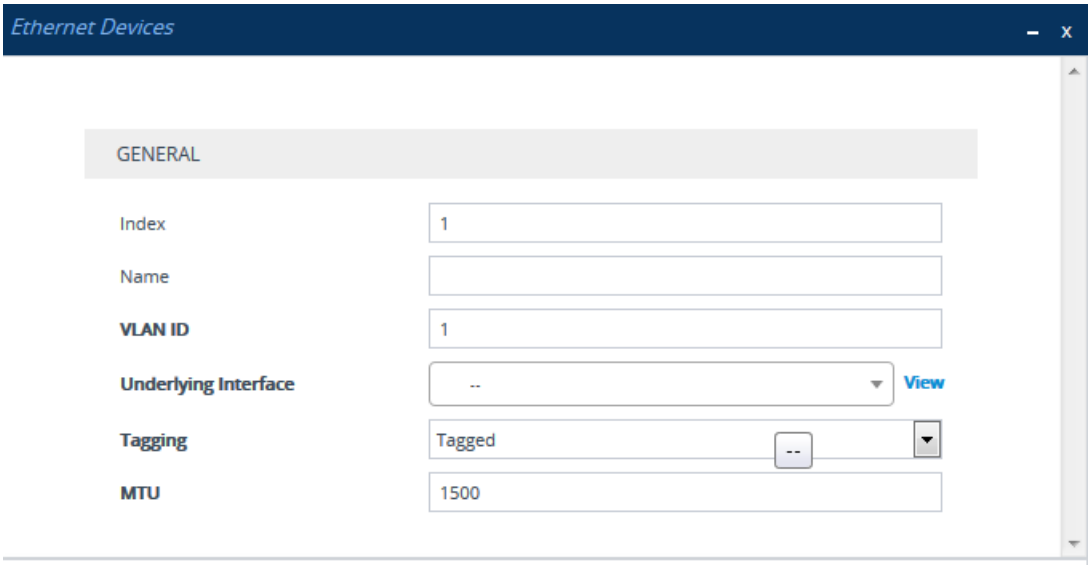
You can view configured Ethernet Devices that have been successfully applied to the device (saved to flash) in the Ethernet Device Status table. This page is accessed by clicking the **Ethernet Device Status Table** button located at the bottom of the Ethernet Devices table. The Ethernet Device Status table can also be accessed from the Navigation tree (see Viewing Ethernet Device Status).

You cannot delete an Ethernet Device that is associated with an IP network interface (in the IP Interfaces table). You can only delete it once you have disassociated it from the IP network interface.

The following procedure describes how to configure Ethernet Devices through the Web interface. You can also configure it through ini file [DeviceTable] or CLI (`configure network > network-dev`).

➢ **To configure an Ethernet Device:**

1. Open the Ethernet Devices table (**Setup** menu > **IP Network** tab > **Core Entities** folder > **Ethernet Devices**).
2. Click **New**; the following dialog box appears:



3. Configure an Ethernet Device according to the parameters described in the table below.
4. Click **Apply**.

Table 13-4: Ethernet Devices Table Parameter Descriptions

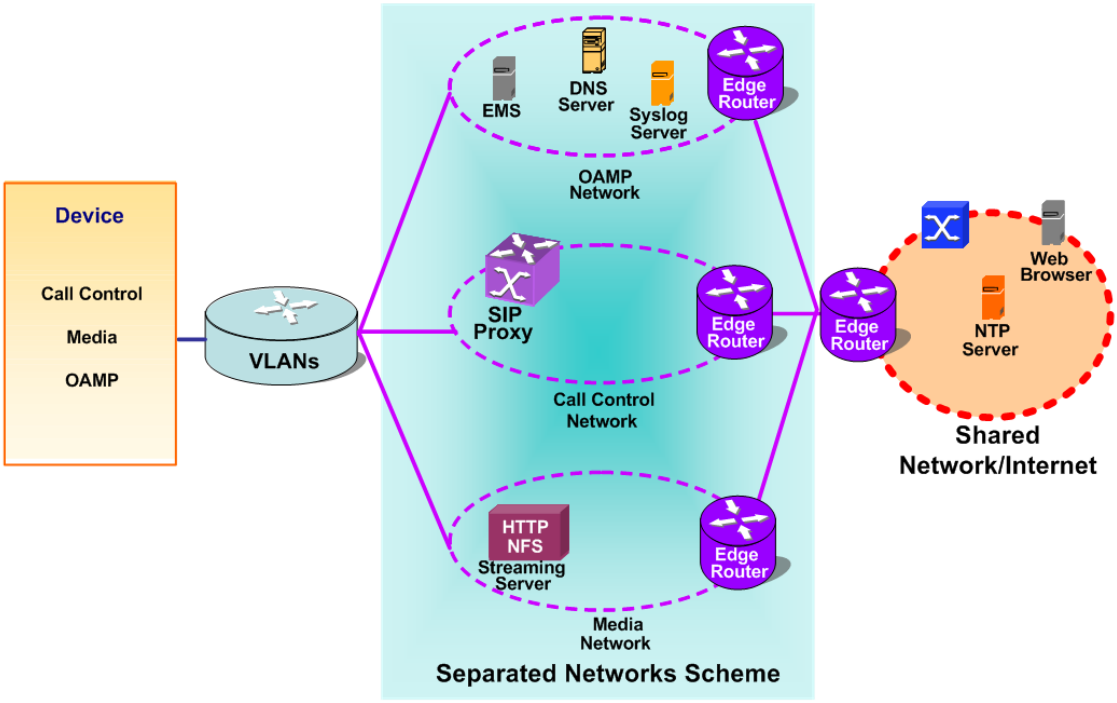
Parameter	Description
'Index' [DeviceTable_Index]	Defines an index number for the new table row. **Note:** Each row must be configured with a unique index.
'Name' `name` [DeviceTable_ DeviceName]	Defines a name for the Ethernet Device. The name is used to associate the Ethernet Device with an IP network interface in the IP Interfaces table ('Underlying Device' field - see Configuring IP Network Interfaces) and/or with a static route in the Static Routes table ('Ethernet Output Device' field - see Configuring Static IP Routing).
'VLAN ID' `vlan-id` [DeviceTable_VlanID]	Defines a VLAN ID for the Ethernet Device. The valid value is 1 to 3999. The default is 1. **Note:** Each Ethernet Device must be configured with a unique VLAN ID.
'Underlying Interface' `underlying-if` [DeviceTable_ UnderlyingInterface]	Assigns an Ethernet Group to the Ethernet Device. To configure Ethernet Groups, see Configuring Ethernet Port Groups. **Note:** The parameter is mandatory.
'Tagging' `tagging` [DeviceTable_Tagging]	Defines VLAN tagging for the Ethernet Device. ■ [0] **Untagged** = (Default for pre-configured Ethernet Device) The Ethernet Device accepts untagged packets and packets with the same VLAN ID as the Ethernet Device. Incoming

Parameter	Description
	untagged packets are assigned the VLAN ID of the Ethernet Device. The Ethernet Device sends these VLAN packets untagged (i.e., removes the VLAN ID). ■ [1] **Tagged** = (Default for new Ethernet Devices) The Ethernet Device accepts packets that have the same VLAN ID as the Ethernet Device and sends packets with this VLAN ID. For all Ethernet Devices that are associated with the same Ethernet Group (see 'Underlying Interface' parameter above) and configured to **Tagged**, incoming untagged packets received on this Ethernet Group are discarded. **Note:** Only one Ethernet Device can be configured as Untagged per associated Ethernet Group. In other words, if multiple Ethernet Devices are associated with the same Ethernet Group, only one of these Ethernet Devices can be configured to **Untagged;** all the others must be configured to **Tagged**.
'MTU' `mtu` [DeviceTable_MTU]	Defines the Maximum Transmission Unit (MTU) in bytes per VLAN (Ethernet Device). The valid value is 68 to 1,500. The default is 1,500. **Note:** ■ MTU is not applicable to SBC Direct Media traffic and to debug recording traffic. ■ If your first Ethernet Device is configured with an untagged VLAN, its MTU value is the maximum MTU that can be configured for all other Ethernet Devices that are associated with the same Ethernet Group. In other words, if you configure additional Ethernet Devices (tagged VLANs) that are associated with the same Ethernet Group, their MTUs must be equal to or less than the MTU of the first Ethernet Device (untagged VLAN). For example, if the untagged VLAN is configured with MTU of 100 bytes, you can configure a tagged VLAN with an MTU value of either 100 bytes or less. ■ If your first Ethernet Device is configured with a tagged VLAN and you later configure an additional Ethernet Device with an untagged VLAN that is associated with the same Ethernet Group, the MTU of the untagged VLAN must be equal to or greater than the highest MTU value configured out of all the Ethernet Devices (VLANs) associated with the Ethernet Group. For example, if VLAN 1 is configured with the highest MTU (100 bytes) out of all your VLANs, you can

Parameter	Description
	configure an untagged VLAN with an MTU value of either 100 bytes or greater.

Configuring IP Network Interfaces

You can configure a single VoIP network interface for all applications, including OAMP (management traffic), call control (SIP signaling messages), and media (RTP traffic), or you can configure multiple logical, IP network interfaces for these applications. You may need to logically separate network segments for these applications for administration and security. This can be achieved by employing Layer-2 VLANs and Layer-3 subnets. The figure below illustrates a typical network architecture where the device is configured with three network interfaces, each representing the OAMP, call control, and media applications. The device is connected to a VLAN-aware switch for directing traffic from and to the device to the three separated Layer-3 broadcast domains according to VLAN tags (middle pane).



The device is shipped with a default OAMP interface (see Default IP Address). The IP Interfaces table lets you change this OAMP interface and configure additional network interfaces for control and media, if necessary. You can configure up to 1,024 interfaces, consisting of up to 1,023 Control and Media interfaces including a Maintenance interface if your device is deployed in a High Availability (HA) mode, and 1 OAMP interface. Each IP interface is configured with the following:

- Application type allowed on the interface:
  - Control: call control signaling traffic (i.e., SIP)
  - Media: RTP traffic

- Operations, Administration, Maintenance and Provisioning (OAMP): management (i.e., Web, CLI, and SNMP based management)
- Maintenance: This interface is used in HA mode when two devices are deployed for redundancy, and represents one of the LAN interfaces or Ethernet Groups on each device used for the Ethernet connectivity between the two devices. For more information on HA and the Maintenance interface, see Configuring High Availability.

- IP address (IPv4 or IPv6) and subnet mask (prefix length)
- Default Gateway: Traffic from this interface destined to a subnet that does not meet any of the routing rules (local or static) are forwarded to this gateway.
- (Optional) Primary and secondary domain name server (DNS) addresses for resolving FQDNs into IP addresses.
- Ethernet Device: Layer-2 bridging device and assigned a VLAN ID. As the Ethernet Device is associated with an Ethernet Group, this is useful for setting trusted and un-trusted networks on different physical Ethernet ports. Multiple entries in the IP Interfaces table may be associated with the same Ethernet Device, providing multi-homing IP configuration (i.e., multiple IP addresses on the same interface/VLAN).

Complementing the IP Interfaces table is the Static Routes table, which lets you configure static routing rules for non-local hosts/subnets. For more information, see Configuring Static IP Routing.

Before configuring the IP network interfaces, read the following guidelines:

- **One** OAMP IP interface must be configured and this **must** be an IPv4 address. The OAMP interface can be combined with Media and Control.
- At least one Media-type interface **must** be configured.
- At least one Control-type interface **must** be configured.
- Multiple Control- and Media-type IP interfaces can be configured with overlapping IP addresses and subnets.
- The prefix length replaces the dotted-decimal subnet mask presentation and **must** have a value of 0-30 for IPv4 addresses, and a value of 64 for IPv6 addresses.
- Multiple Media- and Control-type interfaces can be configured with an IPv6 address.
- IP interface types can be combined:
  - Example 1:
    - One combined OAMP-Media-Control interface with an IPv4 address
  - Example 2:
    - One OAMP-type interface with an IPv4 address
    - One or more Control-type interfaces with IPv4 addresses
    - One or more Media-type interfaces with IPv4 interfaces

- Example 3:
  - One OAMP-type with an IPv4 address
  - One combined Media-Control-type interface with IPv4 address
  - One combined Media-Control-type interface with IPv6 address

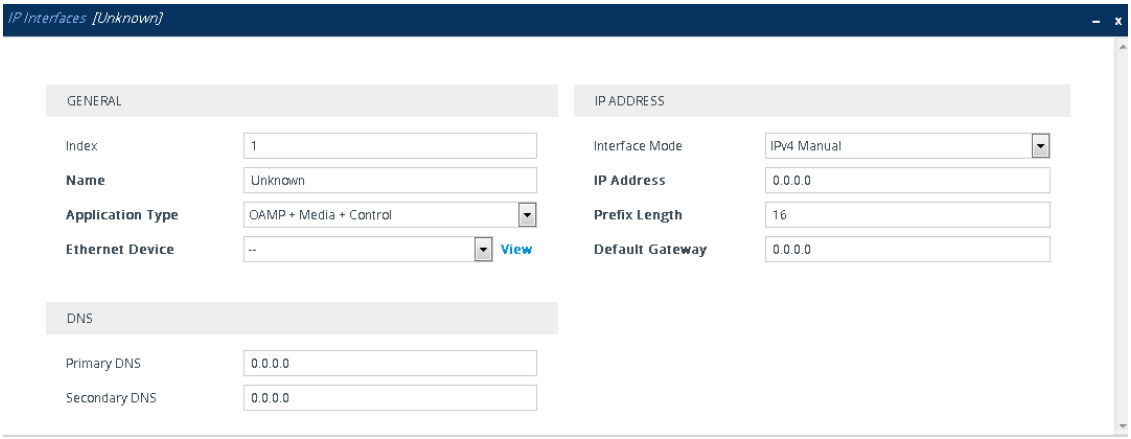
- Multiple IP interfaces that are assigned the same Ethernet Device can't be configured with different Default Gateways. If you need to use a different Default Gateway for one of the subnets defined on this Ethernet Device, to get to some specific network (and not a default destination route), configure a Static Route rule.
- The address of the Default Gateway **must** be in the same subnet as the associated interface. Additional static routing rules can be configured in the Static Routes table.
- The interface name **must** be configured (mandatory) and must be unique for each interface.
- Each network interface must be assigned an Ethernet Device.
- For IPv4 addresses, the 'Interface Mode' column must be set to IPv4 Manual. For IPv6 addresses, this column must be set to IPv6 Manual or IPv6 Manual Prefix.

- Upon device start up, the IP Interfaces table is parsed and passes A comprehensive validation test. If any errors occur during this validation phase, the device sends an error message to the Syslog server and falls back to a "safe mode", using a single interface without VLANs. Ensure that you view the Syslog messages that the device sends in system startup to see if any errors occurred.
- You can associate the Network Time Protocol (NTP) application with the OAMP or Control type IP network interface, using the EnableNTPasOAM ini file parameter. For more information on NTP, see Configuring Automatic Date and Time using SNTP.

The following procedure describes how to configure IP network interfaces through the Web interface. You can also configure it through ini file [InterfaceTable] or CLI (`configure network > interface network-if`).

➢ **To configure IP network interfaces:**

1. Open the IP Interfaces table (**Setup** menu > **IP Network** tab > **Core Entities** folder > **IP Interfaces**).
2. Click **New**; the following dialog box appears:



3. Configure the IP network interface according to the parameters described in the table below.
4. Click **Apply**.

- When you change the device's OAMP interface address and then click **Apply**, connectivity with the device is lost. You need to re-access the device using the new address, and then click the **Save** button on the toolbar for the new address to take effect.
- If you edit or delete an IP interface, current calls using the interface are immediately terminated.
- If you delete an IP interface, rows configured in other tables (e.g., Media Realms table) that are associated with the deleted IP interface, lose their association with the IP interface ('Interface Name' field displays "None") and the rows become invalid.
- If you edit or delete the Maintenance interface (for HA mode), you must reset the device for your changes to take effect.

To view configured IP network interfaces that are currently active, click the **IP Interface Status Table** link located at the bottom of the table. For more information, see Viewing Active IP Interfaces.

Table 13-5: IP Interfaces Table Parameters Description

Parameter	Description
General	
'Index' `network-if` [InterfaceTable_Index]	Defines an index number for the new table row. **Note:** Each row must be configured with a unique index.
'Name' `name` [InterfaceTable_InterfaceName]	Defines a name for the interface. The valid value is a string of up to 16 characters. The default (if no name is configured) is "Interface_n", where *n* is the row index number.

Parameter	Description
	Note: ■ Each row must be configured with a unique name. ■ The parameter value cannot contain a forward slash (/).
'Application Type' `application-type` [InterfaceTable_ApplicationTypes]	Defines the applications allowed on the IP interface. ■ [0] **OAMP** = Operations, Administration, Maintenance and Provisioning (OAMP) applications (e.g., Web, Telnet, SSH, and SNMP). ■ [1] **Media** = Media (i.e., RTP streams of voice). ■ [2] **Control** = Call Control applications (e.g., SIP). ■ [3] **OAMP + Media** = OAMP and Media applications. ■ [4] **OAMP + Control** = OAMP and Call Control applications. ■ [5] **Media + Control** = Media and Call Control applications. ■ [6] **OAMP + Media + Control** = All application types are allowed on the interface. . ■ [99] **MAINTENANCE** = Only the Maintenance application for HA is allowed on this interface. **Note:** ■ Only one IP network interface can be configured with OAMP in this table. To configure additional management interfaces, see Configuring Additional Management Interfaces.
'Ethernet Device' `underlying-dev` [InterfaceTable_ UnderlyingDevice]	Assigns an Ethernet Device to the IP interface. An Ethernet Device is a VLAN associated with a physical Ethernet port (Ethernet Group). To configure Ethernet Devices, see Configuring Underlying Ethernet Devices. By default, no value is defined. Note: The parameter is mandatory.
IP Address	

Parameter	Description
'Interface Mode' `mode` [InterfaceTable_InterfaceMode]	Defines the method that the interface uses to acquire its IP address. ■ [3] **IPv6 Manual Prefix** = IPv6 manual prefix IP address assignment. The IPv6 prefix (higher 64 bits) is set manually while the interface ID (the lower 64 bits) is derived from the device's MAC address. ■ [4] **IPv6 Manual** = IPv6 manual IP address (128 bits) assignment. ■ [10] **IPv4 Manual** = (Default) IPv4 manual IP address (32 bits) assignment.
'IP Address' `ip-address` [InterfaceTable_IPAddress]	Defines an IP address. The valid value is an IPv4 address (in dotted-decimal notation) or an IPv6 address (see RFC 4291). By default, no value is defined. **Note:** The parameter is mandatory.
'Prefix Length' `prefix-length` [InterfaceTable_PrefixLength]	Defines the prefix length of the related IP address. This is a Classless Inter-Domain Routing (CIDR)-style representation of a dotted-decimal subnet notation. The CIDR-style representation uses a suffix indicating the number of bits which are set in the dotted-decimal format. For example, 192.168.0.0/16 is synonymous with 192.168.0.0 and subnet 255.255.0.0. This CIDR lists the number of '1' bits in the subnet mask (i.e., replaces the standard dotted-decimal representation of the subnet mask for IPv4 interfaces). For example, a subnet mask of 255.0.0.0 is represented by a prefix length of 8 (i.e., 11111111 00000000 00000000 00000000) and a subnet mask of 255.255.255.252 is represented by a prefix length of 30 (i.e., 11111111 11111111 11111111 11111100). The prefix length is a Classless Inter-Domain Routing (CIDR) style presentation of a dotted-decimal subnet notation. The CIDR-style presentation is the latest method for interpretation of IP addresses. Specifically, instead of using eight-bit address blocks, it uses the variable-length subnet masking technique to allow allocation on arbitrary-length prefixes. The valid value of the prefix length depends on the IP

Parameter	Description
	address version: ■ IPv4: 0 to 30 ■ IPv6: Depends on the settings of the 'Interface Mode' parameter (above): ✓ **IPv6 Manual Prefix:** 64 ✓ **IPv6 Manual**: Up to 126
'Default Gateway' `gateway` [InterfaceTable_Gateway]	Defines the IP address of the default gateway for the IP interface. When traffic is sent from this interface to an unknown destination (i.e., not in the same subnet and not defined for any static routing rule), it is forwarded to this default gateway. By default, no value is defined.
DNS	
'Primary DNS' `primary-dns` [InterfaceTable_ PrimaryDNSServerIPAddress]	Defines the primary DNS server's IP address (in dotted-decimal notation), which is used for translating domain names into IP addresses for the interface. By default, no IP address is defined.
'Secondary DNS' `secondary-dns` [InterfaceTable_ SecondaryDNSServerIPAddress]	Defines the secondary DNS server's IP address (in dotted-decimal notation), which is used for translating domain names into IP addresses for the interface. By default, no IP address is defined.

Networking Configuration Examples

Examples of IP network interface configuration are listed below:

■ **Single IP network interface for all applications:**

The IP Interfaces table is configured with a single interface for OAMP, Media and Control:

Index	Name	Application Type	Ethernet Device	Interface Mode	IP Address	Prefix Length	Default Gateway
0	myInterface	OAMP + Media + Control	1	IPv4 Manual	192.168.0.2	16	192.168.0.1

Two routes are configured in the **Static Routes table** for directing traffic for subnet 201.201.0.0/16 to 192.168.11.10, and all traffic for subnet 202.202.0.0/16 to 192.168.11.1:

Index	Destination	Prefix Length	Gateway
0	201.201.0.0	16	192.168.11.10
1	202.202.0.0	16	192.168.11.1

The NTP applications remain with their default application types.

- **Multiple interfaces, one per application type:**

The IP Interfaces table is configured with three interfaces, each for a different application type (one for OAMP, one for Call Control, and one for RTP Media), and each with a different VLAN ID and default gateway:

Index	Name	Application Type	Ethernet Device	Interface Mode	IP Address	Prefix Length	Default Gateway
0	ManagementIF	OAMP	1	IPv4 Manual	192.168.0.2	16	192.168.0.1
1	myControlIF	Control	200	IPv4 Manual	200.200.85.14	24	200.200.85.1
2	myMediaIF	Media	211	IPv4 Manual	211.211.85.14	24	211.211.85.1

A routing rule in the Static Routes table is required to allow remote management from a host in 176.85.49.0/24:

Index	Destination	Prefix Length	Gateway
0	176.85.49.0	24	192.168.11.1

All other parameters are set to their respective default values. The NTP application remains with its default application types.

- **Multiple interfaces with combined application types:**

  - A single interface for OAMP.
  - Interfaces for Call Control and Media applications, where two of them are IPv4 interfaces and one is an IPv6 interface.

The IP Interfaces table is configured as follows:

Index	Name	Application Type	Ethernet Device	Interface Mode	IP Address	Prefix Length	Default Gateway
0	Mgmt	OAMP	1	IPv4 Manual	192.168.0.2	16	192.168.0.1
1	MediaCntrl1	Media + Control	201	IPv4 Manual	200.200.85.14	24	200.200.85.1
2	MediaCntrl2	Media + Control	202	IPv4 Manual	200.200.86.14	24	200.200.86.1
3	V6CntrlMedia2	Media + Control	202	IPv6 Manual	2000::1:200:200:86:14	64	::

1. A routing rule in the Static Routes table is required to allow remote management from a host in 176.85.49.0/24:

Index	Destination	Prefix Length	Gateway
0	176.85.49.0	24	192.168.0.10

The NTP application is configured (through the ini file) to serve as OAMP applications:

```
EnableNTPasOAM = 1
```

Configure Layer-2 QoS mapping in the QoS Mapping table. Packets sent with the configured DiffServ get the configured VLAN priority:

Index	Differentiated Services	VLAN Priority
0	46	6
1	40	6
2	26	4
3	10	2

Configure Layer-3 QoS in the QoS Settings table:

- 'Media Premium QoS' - the default DiffServ value is 46

- 'Control Premium QoS' - the default DiffServ value is 24
- 'Gold QoS' - the default DiffServ value is 26
- 'Bronze QoS' - the default DiffServ value is 10

IP Network Interfaces and Multiple Default Gateways:

This example includes a different Default Gateway per IP network interface. The Default Gateway of the OAMP interface is 192.168.0.1 and of the Media and Control is 200.200.85.1. The configuration in the IP Interfaces table is shown below:

Index	Name	Application Type	Ethernet Device	Interface Mode	IP Address	Prefix Length	Default Gateway
0	Mgmt	OAMP	100	IPv4 Manual	192.168.0.2	16	192.168.0.1
1	CntrlMedia	Media & Control	200	IPv4 Manual	200.200.85.14	24	200.200.85.1

Configuring the following static routing rules in the Static Routes table enables OAMP applications to access peers on subnet 17.17.0.0 through the Default Gateway 192.168.10.1 and Media + Control applications to access peers on subnet 171.79.39.0 through Default Gateway 200.200.85.10 (which is not the default gateway of the interface).

Index	Destination	Prefix Length	Ethernet Output Device	Gateway
0	17.17.0.0	16	100	192.168.10.1
1	171.79.39.0	24	200	200.200.85.10

Configuring Static IP Routes

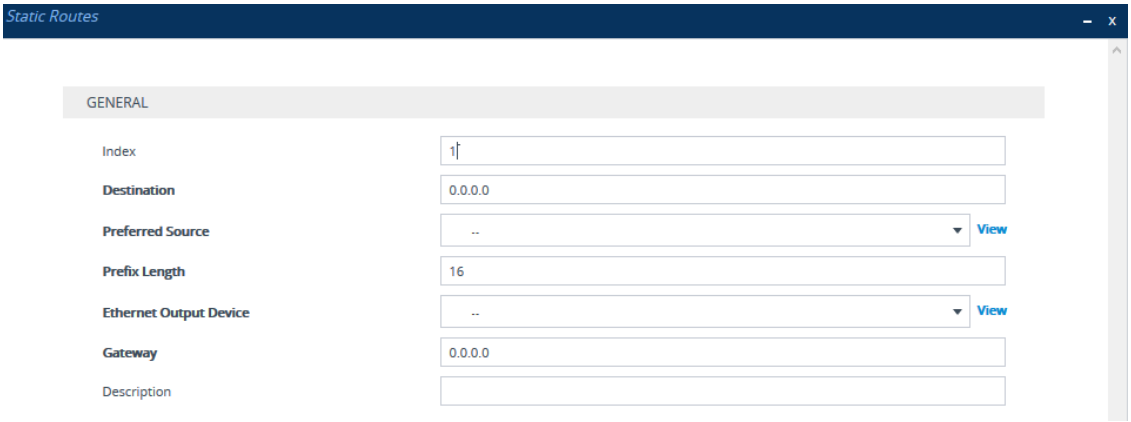
The Static Routes table lets you configure up to 30 static IP routing rules. Static routes let you communicate with LAN networks that are not located behind the Default Gateway that is specified for an IP network interface in the IP Interfaces table, from which the packets are sent. Before sending an IP packet, the device searches the Static Routes table for an entry that matches the requested destination host/network. If an entry is found, the device sends the packet to the gateway that is configured for the static route. If no explicit entry is found, the packet is sent to the Default Gateway as configured for the IP interface in the IP Interfaces table.

You can view the status of configured static routes in the IP Routing Status table. This table can be accessed by clicking the **Static Routes Status Table** link located at the bottom of the Static Routes table (see Viewing Static Routes Status).

The following procedure describes how to configure static routes through the Web interface. You can also configure it through ini file [StaticRouteTable] or CLI (`configure network > static`).

➢ **To configure static IP routes:**

1. Open the Static Routes table (**Setup** menu > **IP Network** tab > **Core Entities** folder > **Static Routes**).
2. Click **New**; the following dialog box appears:



3. Configure a static route according to the parameters described in the table below. The address of the host/network you want to reach is determined by an AND operation that is applied to the fields 'Destination' and 'Prefix Length'. For example, to reach network 10.8.x.x, enter "10.8.0.0" in the 'Destination' field and "16" in the 'Prefix Length'. As a result of the AND operation, the value of the last two octets in the 'Destination' field are ignored. To reach a specific host, enter its IP address in the 'Destination' field and "32" in the 'Prefix Length' field.
4. Click **Apply**, and then save your settings to flash memory.

- You can only delete static routing rules that are inactive.
- You can configure only one Static Route rule with the same 'Destination' and 'Ethernet Output Device'.

Table 13-6: Static Routes Table Parameter Descriptions

Parameter	Description
'Index' [StaticRouteTable_Index]	Defines an index number for the new table row. The valid value is 0 to 29. **Note:** Each row must be configured with a unique index.
'Destination' `destination`	Defines the IP address of the destination host/network. The destination can be a single

Parameter	Description
[StaticRouteTable_Destination]	host or a whole subnet, depending on the prefix length configured for this routing rule.
'Preferred Source' `preferred-source-interface-name` [StaticRouteTable_ PreferredSourceInterfaceName]	Defines a specific local source IP address for outgoing packets using the static route. This is done by assigning the rule a local source IP interface (from the IP Interfaces table, described in Configuring IP Network Interfaces). This parameter can be used when the device sends packets to a specific destination that requires a specific source address, for example, when using multi-homing (multiple IP addresses configured on the same VLAN device). This feature also provides predictability and consistency of locally-generated traffic, which is useful (or even needed) for firewalls, monitoring or reporting, and various other use cases. If the parameter is not configured (default), the device sets the source address of the outgoing packet to the address of the IP Interface that is associated with the static route's VLAN (see the 'Ethernet Output Device' parameter below).
'Prefix Length' `prefix-length` [StaticRouteTable_PrefixLength]	Defines the Classless Inter-Domain Routing (CIDR)-style representation of a dotted-decimal subnet notation of the destination host/network. The CIDR-style representation uses a suffix indicating the number of bits that are set in the dotted-decimal format. For example, the value 16 represents subnet 255.255.0.0 The valid value depends on the IP address version: ■ IPv4: 0 to 32 ■ IPv6: 0 to 128
'Ethernet Output Device' `device-name` [StaticRouteTable_DeviceName]	Associates an IP network interface through which the static route's Gateway is reached. The association is done by assigning the parameter the same Ethernet Device that is assigned to the IP network interface in the IP Interfaces table ('Ethernet Device' parameter). To configure IP network interface, see Configuring IP Network

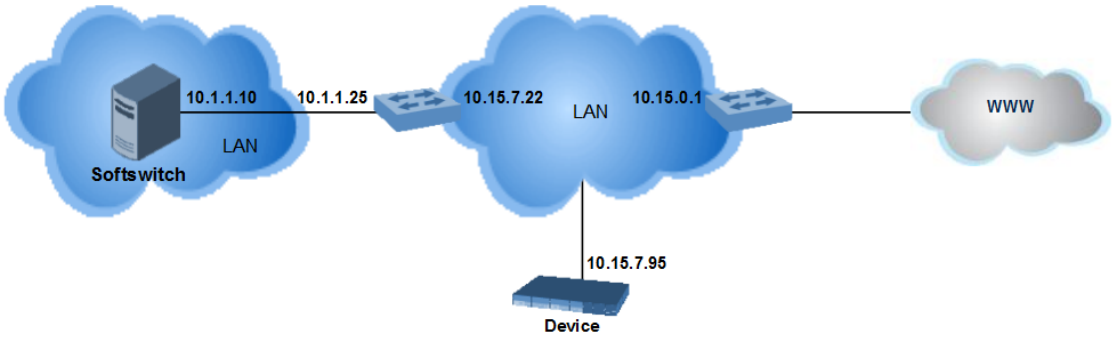
Parameter	Description
	Interfaces. To configure Ethernet Devices, see Configuring Underlying Ethernet Devices.
'Gateway' gateway [StaticRouteTable_Gateway]	Defines the IP address of the Gateway (next hop) used for traffic destined to the subnet/host defined in the 'Destination' / 'Prefix Length' field. **Note:** ■ The Gateway's address must be in the same subnet as the IP address of the network interface that is associated with the static route (using the 'Ethernet Output Device' parameter - see above). ■ The IP network interface associated with the static route must be of the same IP address family (IPv4 or IPv6).
'Description' description [StaticRouteTable_Description]	Defines a name for the rule. The valid value is a string of up to 20 characters.

Configuration Example of Static IP Routes

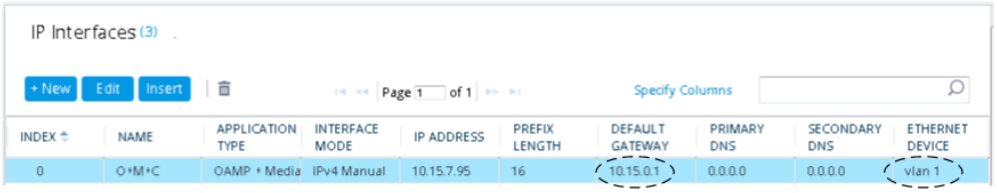
An example of the use for static routes is shown in the figure below. In the example, the device needs to communicate with a softswitch at IP address 10.1.1.10. However, the IP network interface from which packets destined for 10.1.1.10 is sent, is configured to send the packets to a Default Gateway at 10.15.0.1. Therefore, the packets do not reach the softswitch. To resolve this problem, a static route is configured to specify the correct gateway (10.15.7.22) in order to reach the softswitch.

Note the following configuration:

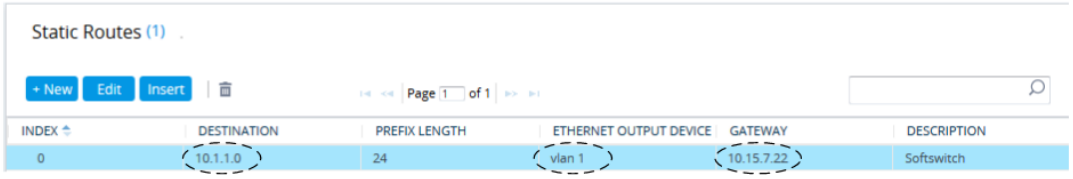
- The static route is configured with a subnet mask of 24 (255.255.255.0), enabling the device to use the static route to send all packets destined for 10.1.1.**x** to this gateway and therefore, to the network in which the softswitch resides.
- The static route in the Static Routes table must be associated with the IP network interface in the IP Interfaces table. This is done by configuring the 'Ethernet Output Device' field in the Static Routes table to the same value as configured in the 'Ethernet Device' field in the IP Interfaces table.
- The static route's Gateway address in the Static Routes table is in the same subnet as the IP address of the IP network interface in the IP Interfaces table.



No Static Route:
The device sends packets to 10.15.0.1, which is the Default Gateway defined for this IP network interface in the IP Interfaces table. Therefore, the device will not succeed in reaching the softswitch.



Static Route Configured:
A static route with the correct gateway is needed for routing to the softswitch. The device communicates with the softswitch (10.1.1.0/24) using the gateway 10.15.7.22. Note that the device first searches for a matching route in the Static Routes table.
If not found, it uses the default gateway defined in the IP Interfaces table.



Troubleshooting the Static Routes Table

When adding a new static route to the Static Routes table, the added rule passes a validation test. If errors are found, the static route is rejected and not added to the table. Failed static route validations may result in limited connectivity (or no connectivity) to the destinations specified in the incorrect static route. For any error found in the Static Routes table or failure to configure a static route, the device sends a notification message to the Syslog server reporting the problem.

Common static routing configuration errors may include the following:

- The IP address specified in the 'Gateway' field is unreachable from the IP network interface associated with the static route.
- The same destination has been configured in two different static routing rules.
- More than 30 static routes have been configured.

> If a static route is required to access OAMP applications (for remote management, for example) and the route is not configured correctly, the route is not added and the device is not accessible remotely. To restore connectivity, the device must be accessed locally from the OAMP subnet and the required routes be configured.

Network Address Translation Support

Network Address Translation (NAT) is a mechanism that maps internal IP addresses (and ports) used within a private network to global IP addresses and vice versa, providing transparent routing to end hosts. The primary advantages of NAT include (1) reduction in the number of global IP addresses required in a private network (global IP addresses are only used to connect to the Internet) and (2) better network security by hiding the internal architecture.

The design of SIP creates a problem for VoIP traffic to pass through NAT. SIP uses IP addresses and port numbers in its message body. However, the NAT server is unable to modify the SIP messages and thus, can't change local addresses to global addresses.

This section discusses the device's solutions for overcoming NAT traversal issues.

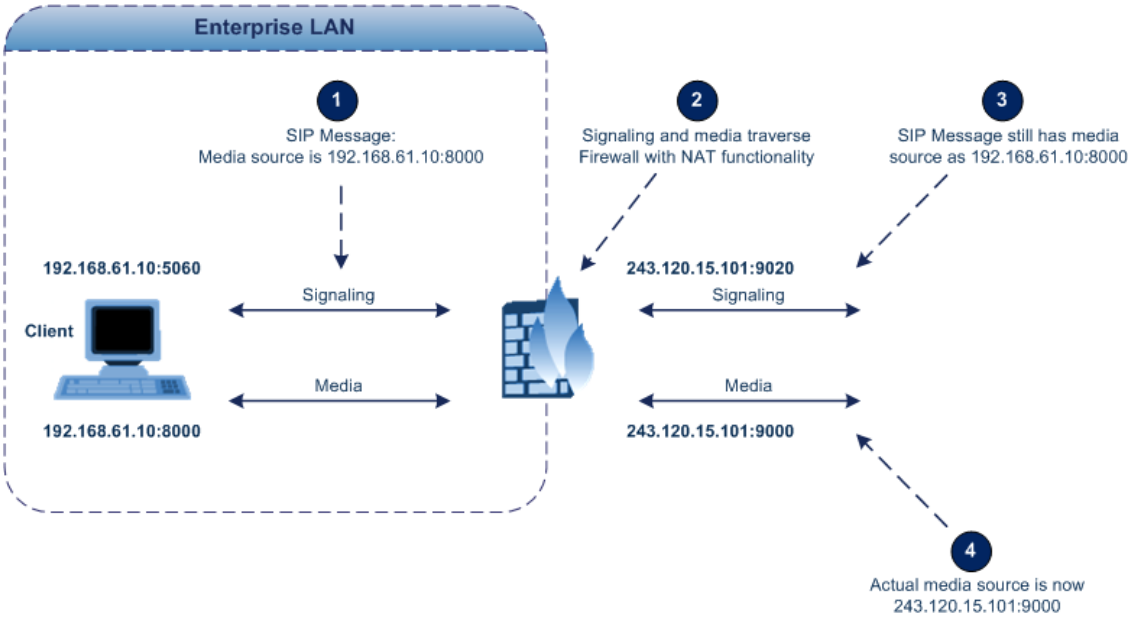
Device Located behind NAT

Two different streams of traffic traverse through NAT - signaling and media. A device located behind NAT that initiates a signaling path has problems receiving incoming signaling responses, as they are blocked by the NAT server. Therefore, the initiating device must inform the receiving device where to send the media. To resolve this NAT problem, the device provides the following solution:

- NAT Translation table, which configures NAT per IP network interface - see Configuring NAT Translation per IP Interface.

If NAT is not configured, the device sends the packet according to its IP address configured in the IP Interfaces table.

The figure below illustrates the NAT problem faced by SIP networks when the device is located behind a NAT:



Configuring NAT Translation per IP Interface

The NAT Translation table lets you configure up to 32 network address translation (NAT) rules for translating source IP addresses into NAT IP addresses (*global* - *public*) when the device is located behind NAT. The device's NAT traversal mechanism replaces the source IP address of SIP messages sent from a specific VoIP interface (Control and/or Media) in the IP Interfaces table to a public IP address. This allows, for example, the separation of VoIP traffic between different ITSPs and topology hiding of internal IP addresses from the "public" network. Each IP network interface (configured in the IP Interfaces table) can be associated with a NAT rule, translating the source IP address and port of the outgoing packet into the NAT address (IP address and port range). For Mediant CE, each remote IP interface for media on the Media Components can be associated with a NAT rule.

The following procedure describes how to configure NAT translation rules through the Web interface. You can also configure it through ini file [NATTranslation] or CLI (`configure network > nat-translation`).

➢ To configure NAT translation rules:

1. Open the NAT Translation table (**Setup** menu > **IP Network** tab > **Core Entities** folder > **NAT Translation**).
2. Click **New**; the following dialog box appears:

3. Configure a NAT translation rule according to the parameters described in the table below.
4. Click **Apply**, and then save your settings to flash memory.

Table 13-7: NAT Translation Table Parameter Descriptions

Parameter	Description
Source	
'Index' `index` [NATTranslation_Index]	Defines an index number for the new table row. **Note:** Each row must be configured with a unique index.
'Source Interface'	Assigns an IP network interface (configured in the IP

Parameter	Description
src-interface-name [NATTranslation_ SrcIPInterfaceName]	Interfaces table) to the rule. Outgoing packets sent from the specified network interface are NAT'ed. By default, no value is defined. To configure IP network interfaces, see Configuring IP Network Interfaces.
'Source Start Port' src-start-port [NATTranslation_ SourceStartPort]	Defines the optional starting port range (0-65535) of the IP interface, used as matching criteria for the NAT rule. If not configured, the match is done on the entire port range. Only IP addresses and ports of matched source ports will be replaced.
'Source End Port' src-end-port [NATTranslation_ SourceEndPort]	Defines the optional ending port range (0-65535) of the IP interface, used as matching criteria for the NAT rule. If not configured, the match is done on the entire port range. Only IP addresses and ports of matched source ports will be replaced.
Target	
'Target IP Address' target-ip-address [NATTranslation_ TargetIPAddress]	Defines the global (public) IP address. The device adds the address in the outgoing packet to the SIP Via header, Contact header, 'o=' SDP field, and 'c=' SDP field.
'Target Start Port' target-start-port [NATTranslation_ TargetStartPort]	Defines the optional starting port range (0-65535) of the global address. If not configured, the ports are not replaced. Matching source ports are replaced with the target ports. This address is set in the SIP Via and Contact headers and in the 'o=' and 'c=' SDP fields.
'Target End Port' target-end-port [NATTranslation_ TargetEndPort]	Defines the optional ending port range (0-65535) of the global address. If not configured, the ports are not replaced. Matching source ports are replaced with the target ports. This address is set in the SIP Via and Contact headers and in the 'o=' and 'c=' SDP fields.

Remote UA behind NAT

This section describes configuration for scenarios where the device sends signaling and media packets to a remote UA that is located behind NAT.

SIP Signaling Messages

By default, the device resolves NAT issues for SIP signaling, using its NAT Detection mechanism. The NAT Detection mechanism checks whether the endpoint is located behind NAT by comparing the source IP address of the incoming UDP/TCP packet (in which the SIP message is received) with the IP address in the SIP Contact header. If the packet's source IP address is a public address and the Contact header's IP address is a local address, the device considers the endpoint as located behind NAT. In this case, the device sends the SIP messages to the endpoint using the packet's source IP address. Otherwise (or if you have disabled the NAT Detection mechanism), the device sends the SIP messages according to the SIP standard (RFC 3261), where requests within the SIP dialog are sent using the IP address in the Contact header and responses to INVITEs are sent using the IP address in the Via header.

If necessary, you can also configure the device to always consider incoming SIP INVITE messages as sent from endpoints that are located behind NAT. When this is enabled, the device sends responses to the INVITE (to the endpoint) using the the source IP address of the packet (INVITE) initially received from the endpoint. This is useful in scenarios where the endpoint is located behind a NAT firewall and the device (for whatever reason) is unable to identify NAT using its regular NAT Detection mechanism. This feature is enabled per specific calls using the 'Always Use Source Address' parameter in the IP Groups table (see Configuring IP Groups). If this feature is disabled, the device's NAT detection is according to the settings of the global parameter, 'SIP NAT Detection' parameter (see below procedure).

➢ **To enable the NAT Detection feature (global):**

1. Open the Transport Settings page (**Setup** menu > **Signaling & Media** tab > **SIP Definitions** folder > **Transport Settings**).
2. From the 'SIP NAT Detection' drop-down list (SIPNatDetection), select **Enable**:

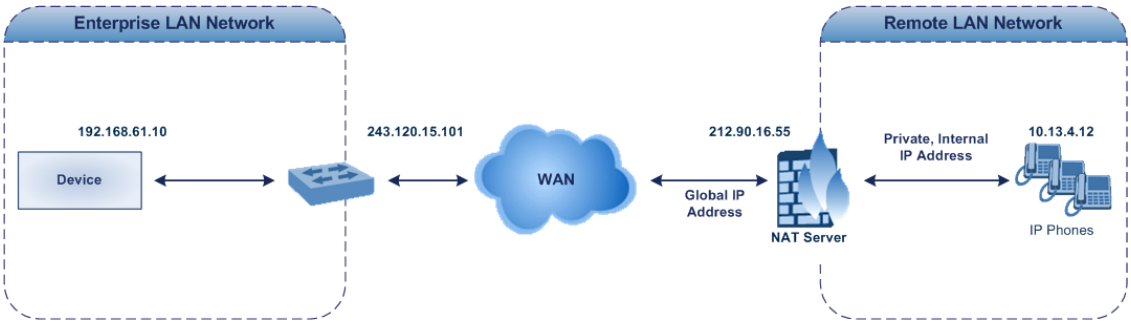
3. Click **Apply**.

Media (RTP/RTCP/T.38)

When a remote UA initiates a call and is not located behind a NAT server, the device sends the media (RTP, RTCP, and T.38) packets to the remote UA using the IP address:port (UDP) indicated in the SDP body of the SIP message received from the UA. However, if the UA is located behind NAT, the device sends the RTP with the IP address of the UA (i.e., private IP address) as the destination instead of that of the NAT server. Thus, the RTP will not reach the UA. To resolve this NAT traversal problem, the device offers the following features:

- First Incoming Packet Mechanism - see First Incoming Packet Mechanism
- RTP No-Op packets according to the avt-rtp-noop draft - see No-Op Packets

The figure below illustrates a typical network architecture where the remote UA is located behind NAT:



First Incoming Packet Mechanism

In scenarios where the remote user agent (UA) resides behind a NAT server, it's possible that the device, if not configured for NAT traversal, will send the media (RTP, RTCP and T.38) streams to an invalid IP address and UDP port. In other words, it will send the media to the private IP address:port of the UA and not the public address (of the NAT server) and therefore, the media will not reach the UA. When the UA is located behind NAT, although the UA sends its private IP address:port in the original SIP message (INVITE), the device receives the media packets with a source address of a public IP address:port (i.e., allocated by the NAT server). Therefore, to ensure that the media reaches the UA, the device must send it to the public address.

The device identifies whether the UA is located behind NAT by comparing the source IP address of the first received media packet with the IP address and UDP port of the first received SIP message (INVITE) when the SIP session was started. This is done for each media type--RTP, RTCP and T.38--and therefore, they can have different destination IP addresses and UDP ports than one another.

The device supports various NAT traversal methods, which you can configure using the 'NAT Traversal' (NATMode) parameter. For more information on the different options provided by this parameter, see NAT and STUN Parameters on page 1167.

➢ **To enable NAT resolution using the First Incoming Packet mechanism:**

1. Open the Media Settings page (**Setup** menu > **Signaling & Media** tab > **Media** folder > **Media Settings**).
2. From the 'NAT Traversal' drop-down list (NATMode), select the required NAT option.

3. Click **Apply**.

No-Op Packets

The device can send No-Op packets to verify Real-Time Transport Protocol (RTP) and T.38 connectivity, and to keep NAT bindings and Firewall pinholes open. The No-Op packets can be sent in RTP and T.38 formats:

- **RTP No-Op:** The RTP No-Op support complies with IETF Internet-Draft draft-wing-avt-rtp-noop-03 ("A No-Op Payload Format for RTP"). The IETF document defines a No-Op payload format for RTP. The draft defines the RTP payload type as dynamic. You can configure the payload type as described in the following procedure (default is 120).
- **T.38 No-Op:** T.38 No-Op packets are sent only while a T.38 session is activated. Sent packets are a duplication of the previously sent frame (including duplication of the sequence number).

➢ **To configure the No-Op packet feature:**

1. Enable the feature, using the [NoOpEnable] parameter. You can also enable the feature per IP Profile, using the 'Generate No-Op Packets' parameter (see Configuring IP Profiles on page 519).
2. Configure the interval between each No-Op packet sent by the device during the silence period (i.e., no RTP or T.38 traffic), using the [NoOpInterval] parameter.
3. For RTP No-Op packets, configure the payload type of the No-Op packets, using the [RTPNoOpPayloadType] parameter.

The receipt of No-Op packets is always supported.

Fax Transmission behind NAT

The device supports transmission from fax machines (connected to the device) located inside (behind) NAT. Generally, the firewall blocks T.38 (and other) packets received from the WAN, unless the device behind the NAT sends at least one IP packet from the LAN to the WAN through the firewall. If the firewall blocks T.38 packets sent from the termination IP fax, the fax fails. To overcome this problem, the device sends No-Op ("no-signal") packets to open a pinhole in the NAT for the answering fax machine. The originating fax does not wait for an answer, but immediately starts sending T.38 packets to the terminating fax machine upon receipt of a re-INVITE with T.38 only in the SDP, or T.38 and audio media in the SDP. This feature is configured using the [T38FaxSessionImmediateStart] parameter. The No-Op packet feature is enabled using the [NoOpEnable] and [NoOpInterval] parameters.

ICE Lite

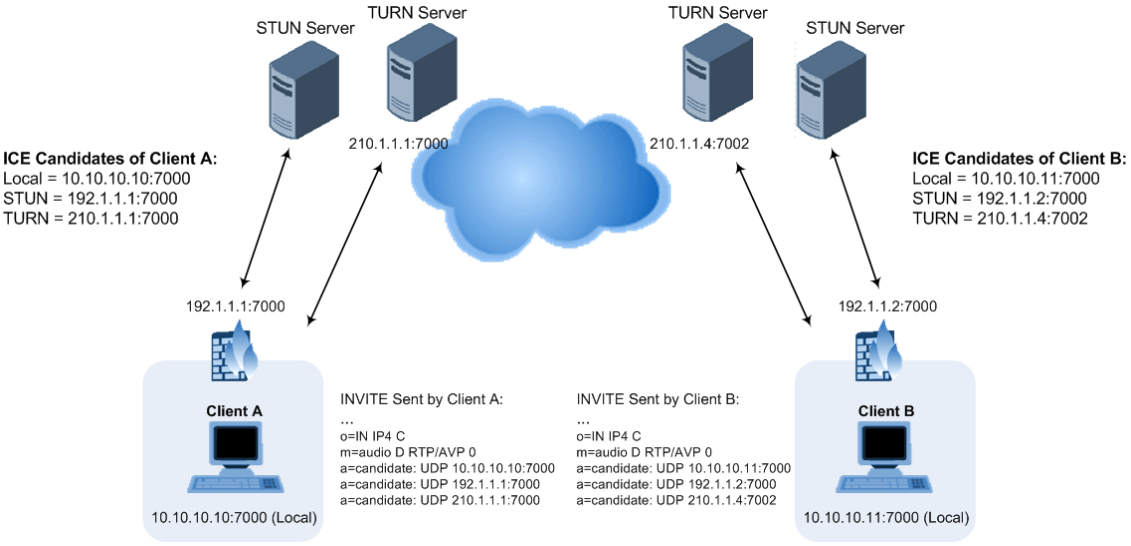
The device supports Interactive Connectivity Establishment (ICE) Lite for SBC calls. ICE is a methodology for NAT traversal, enabling VoIP interoperability across networks to work better across NATs and firewalls. It employs Session Traversal Utilities for NAT (STUN) and Traversal

Using Relays around NAT (TURN) protocols to provide a peer with a public IP address and port that can be used to connect to a remote peer. Therefore, for some applications such as when the device operates in Microsoft Teams Direct Routing (media bypass) environments, ICE Lite is required.

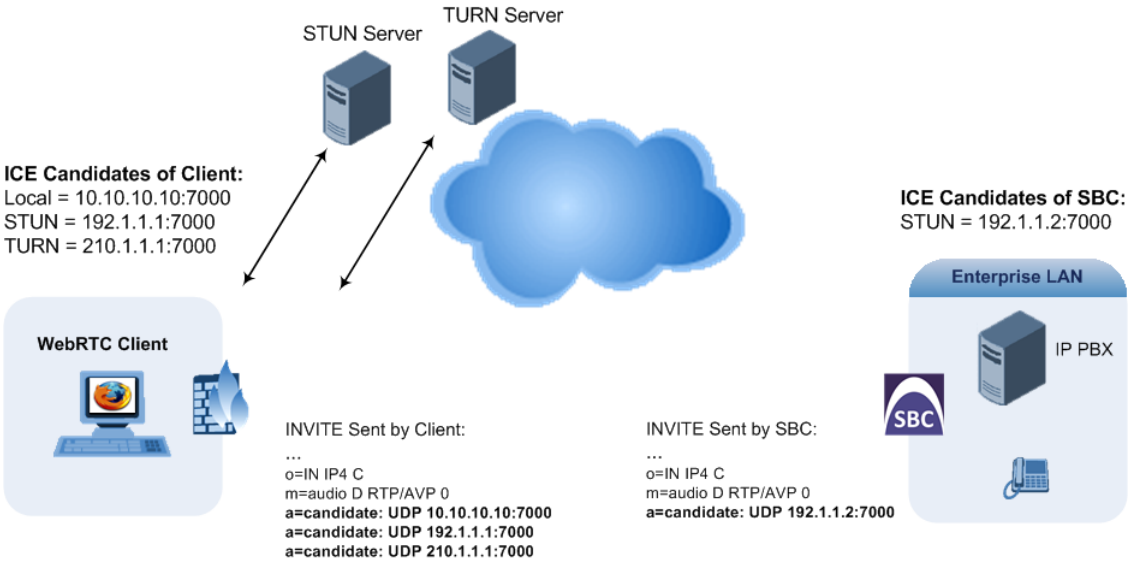
In order for clients behind NATs and/or firewalls to send media (RTP) between one another, they need to discover each others IP address and port as seen by the "outside" world. If both peers are located in different private networks behind a NAT, the peers must coordinate to determine the best communication path between them.

ICE first tries to make a connection using the client's private local address. If that fails (which it will for clients behind NAT), ICE obtains an external (public) address using a STUN server. If that fails, traffic is routed through a TURN relay server (which has a public address).

These addresses:ports (local, STUN, TURN and any other network address) of the client are termed "candidates". Each client sends its' candidates to the other in the SDP body of the INVITE message. Peers then perform connectivity checks per candidate of the other peer, using STUN binding requests sent on the RTP and RTCP ports. ICE tries each candidate and selects the one that works (i.e., media can flow between the clients). The following figure shows a simple illustration of ICE:



The device's support for ICE-Lite means that it does not initiate the ICE process. Instead, it supports remote endpoints that initiate ICE to discover their workable public IP address with the device. Therefore, the device supports the receipt of STUN binding requests for connectivity checks of ICE candidates and responds to them with STUN responses. Note that in the response to the INVITE message received from the remote endpoint, the device sends only a single candidate for its' own IP address. This is the IP address of the device that the client uses. To support ICE, the SBC leg interfacing with the ICE-enabled client (SIP entity) must be enabled for ICE. This is done using the IP Profile parameter [IPProfile_SBCIceMode] - see Configuring IP Profiles.



As the ICE technique has been defined by the WebRTC standard as mandatory for communication with the WebRTC client, ICE support by the device is important for deployments implementing WebRTC. For more information on WebRTC, see WebRTC. Once a WebRTC session (WebSocket) is established for SIP signaling between the device and the WebRTC client, the client's IP address needs to be discovered by the SBC device using the ICE technique.

Robust Receipt of Media Streams by Media Latching

The device's Robust Media feature (or media latching) filters out unwanted media (RTP, RTCP, SRTP, SRTCP, and T.38) streams that are sent to the same port number of the device. Media ports may receive additional multiple unwanted media streams (from multiple sources of traffic) as result of traces of previous calls, call control errors, or deliberate malicious attacks (e.g., Denial of Service). When the device receives more than one media stream on the same port, the Robust Media mechanism detects the valid media stream and ignores the rest. Thus, this can prevent an established call been stolen by a malicious attacker on the media stream.

For the involved voice channel, the device latches on to the first stream of the first received packet. All packets (of any media type) received from the same IP address and SSRC are accepted (for T.38 packets, the device considers only the IP address). If the channel receives subsequent packets from a non-latched source, the device can either ignore this new stream and remain latched to the first original stream (IP address:port) or it can latch on to this new stream. The media latch mode is configured using the InboundMediaLatchMode parameter. If this mode is configured to latch on to new streams, you also need to configure the following:

- Minimum number of continuous media packets that need to be received from a different source(s) before the channel can latch onto this new incoming stream.
- Period (msec) during which if no packets are received from the current stream, the channel latches onto the next packet received from any other stream.

Depending on media latch mode, if the device has latched on to a new stream and a packet from the original (first latched onto) IP address:port is received at any time, the device latches on to this original stream.

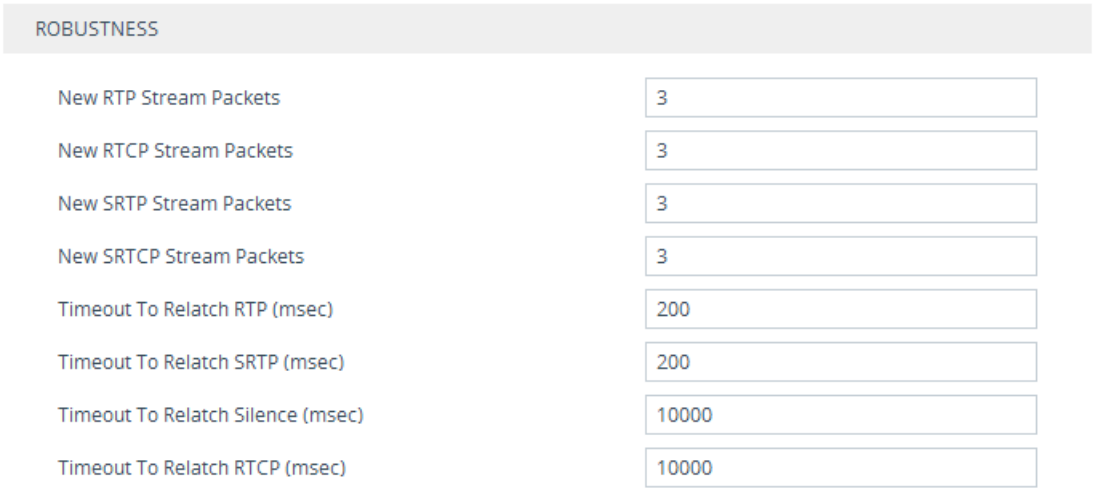
Latching on to a new T.38 stream is reported in CDR using the CDR fields, LatchedT38Ip (new IP address) and LatchedT38Port (new port). In addition, the SIP PUBLISH message updates the latched RTP SSRC, for example:

RemoteAddr: IP=10.33.2.55 Port=4000 SSRC=0x66d510ec

➢ To configure media latching:

1. Open the Media Settings page (**Setup** menu > **Signaling & Media** tab > **Media** folder > **Media Settings**), and then from the 'Inbound Media Latch Mode' drop-down list (InboundMediaLatchMode), configure the media latch mode:

2. If you configure Step 1 to **Dynamic** or **Dynamic-Strict**:
   - Configure the minimum number of continuous media (RTP, RTCP, SRTP, and SRTCP) packets that need to be received by the channel before it can latch onto this new incoming stream:
     - 'New RTP Stream Packets'
     - 'New RTCP Stream Packets'
     - 'New SRTP Stream Packets'
     - 'New SRTCP Stream Packets'
   - Configure a period (msec) after which if no packets are received from the current media session, the channel can re-latch onto another stream:
     - 'Timeout To Relatch RTP'
     - 'Timeout To Relatch SRTP'
     - 'Timeout To Relatch Silence'
     - 'Timeout To Relatch RTCP'
     - 'Fax Relay Rx/Tx Timeout'



3. Click **Apply**, and then save your settings to flash memory.

Configuring Quality of Service

This section describes how to configure Layer-2 and Layer-3 Quality of Service (QoS).

Configuring Class-of-Service QoS

The QoS Settings page lets you configure Layer-3 Class-of-Service Quality of Service (QoS). This configures Differentiated Services (DiffServ) values for each CoS. DiffServ is an architecture providing different types or levels of service for IP traffic. DiffServ (according to RFC 2474), prioritizes certain traffic types based on priority, accomplishing a higher-level QoS at the expense of other traffic types. By prioritizing packets, DiffServ routers can minimize transmission delays for time-sensitive packets such as VoIP packets.

You can assign DiffServ to the following class of services (CoS):

- Media Premium: RTP packets sent to the LAN
- Control Premium: Control protocol (SIP) packets sent to the LAN
- Gold: HTTP streaming packets sent to the LAN
- Bronze: OAMP packets sent to the LAN

The mapping of an application to its CoS and traffic type is shown in the table below:

Table 13-8: Traffic/Network Types and Priority

Application	Traffic / Network Types	Class-of-Service (Priority)
Debugging interface	Management	Bronze
Telnet	Management	Bronze
Web server (HTTP)	Management	Bronze

Application	Traffic / Network Types	Class-of-Service (Priority)
SNMP GET/SET	Management	Bronze
Web server (HTTPS)	Management	Bronze
RTP traffic	Media	Media Premium
RTCP traffic	Media	Media Premium
T.38 traffic	Media	Media Premium
SIP	Control	Control Premium
SIP over TLS (SIPS)	Control	Control Premium
Syslog	Management	Bronze
SNMP Traps	Management	Bronze
DNS client	Varies according to DNS settings: ■ OAMP ■ Control	Depends on traffic type: ■ Control: Control Premium ■ Management: Bronze
NTP	Varies according to the interface type associated with NTP (see Assigning NTP Services to Application Types): ■ OAMP ■ Control	Depends on traffic type: ■ Control: Control Premium ■ Management: Bronze

➢ **To configure DiffServ (Layer-3 QoS) values per CoS:**

1. Open the QoS Settings page (**Setup** menu > **IP Network** tab > **Quality** folder > **QoS Settings**).
2. Click **New**; the following dialog box appears:

3. Configure DiffServ values per CoS according to the parameters described in the table below.
4. Click **Apply**, and then save your settings to flash memory.

Table 13-9: QoS Settings Parameter Descriptions

Parameter	Description
'Media Premium QoS' `media-qos` [PremiumServiceClassMediaDiffServ]	Defines the DiffServ value for Premium Media CoS content. The valid range is 0 to 63. The default is 46. **Note:** You can also configure the the parameter per IP Profile (IpProfile_IPDiffServ) or Tel Profile (TelProfile_IPDiffServ).
'Control Premium QoS' `control-qos` [PremiumServiceClassControlDiffServ]	Defines the DiffServ value for Premium Control CoS content (Call Control applications). The valid range is 0 to 63. The default is 24. **Note:** You can also configure the the parameter per IP Profile (IpProfile_ SigIPDiffServ) or Tel Profile (TelProfile_ SigIPDiffServ).
'Gold QoS' `gold-qos` [GoldServiceClassDiffServ]	Defines the DiffServ value for Gold CoS content (streaming applications). The valid range is 0 to 63. The default is 26.
'Bronze QoS' `bronze-qos` [BronzeServiceClassDiffServ]	Defines the DiffServ value for Bronze CoS content (OAMP applications). The valid range is 0 to 63. The default is 10.

Configuring DiffServ-to-VLAN Priority Mapping

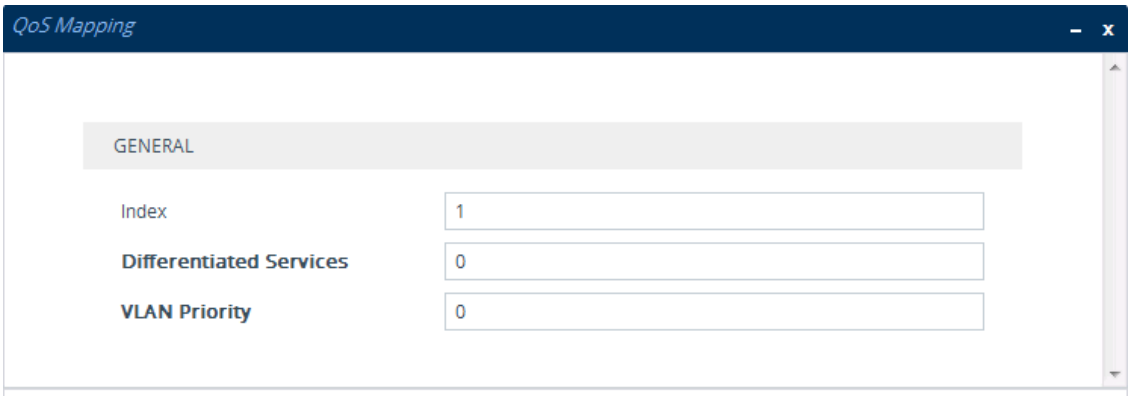
The QoS Mapping table lets you configure up to 64 DiffServ-to-VLAN priority mapping for Layer 3 and Layer-2 Quality of Service (QoS). For each packet sent to the LAN, the VLAN Priority of the packet is set according to the DiffServ value in the IP header of the packet.Layer-2 802.1Q

frames have a 2-byte field called Tag Control Information. The three most significant bits of this 2-byte field represents the Class of Service (CoS) value. Layer-2 QoS is represented by this CoS value which is from 0 to 7 (thus 8 values). Layer-2 QoS parameters define the values for the 3 priority bits in the VLAN tagaccording to the value of the DiffServ field in the packet IP header (according to the IEEE 802.1p standard). Differentiated Services (DiffServ) is an architecture providing different types or levels of service for IP traffic. DiffServ (according to RFC 2474), prioritizes certain traffic types based on priority, accomplishing a higher-level QoS at the expense of other traffic types. By prioritizing packets, DiffServ routers can minimize transmission delays for time-sensitive packets such as VoIP packets.

The following procedure describes how to configure DiffServ-to-VLAN priority mapping through the Web interface. You can also configure it through ini file [DiffServToVlanPriority] or CLI (`configure network > qos vlan-mapping`).

➢ **To configure DiffServ-to-VLAN priority mapping:**

1. Open the QoS Mapping table (**Setup** menu > **IP Network** tab > **Quality** folder > **QoS Mapping**).
2. Click **New**; the following dialog box appears:



3. Configure a DiffServ-to-VLAN priority mapping rule according to the parameters described in the table below.
4. Click **Apply**, and then save your settings to flash memory.

Table 13-10:QoS Mapping Table Parameter Descriptions

Parameter	Description
'Index'	Defines an index number for the new table row. **Note:** Each row must be configured with a unique index.
'Differentiated Services' `diff-serv` [DiffServToVlanPriority_DiffServ]	Defines a DiffServ value. The valid value is 0 to 63. The default is 0.

Parameter	Description
'VLAN Priority' vlan-priority [DiffServToVlanPriority_ VlanPriority]	Defines the VLAN priority level. The valid value is 0 to 7. The default is 0.

Configuring ICMP Messages

Internet Control Message Protocol (ICMP) is one of the core protocols of the Internet Protocol suite. It is used by network devices such as routers to send error messages indicating, for example, that a requested service is unavailable.

You can configure the device to handle ICMP messages as follows:

- Send and receive ICMP Redirect messages.
- Send ICMP Destination Unreachable messages. The device sends this message in response to a packet that cannot be delivered to its destination for reasons other than congestion. The device sends a Destination Unreachable message upon any of the following:
  - Address unreachable
  - Port unreachable

  This feature is applicable to IPv4 and IPv6 addressing schemes.

➢ **To configure handling of ICMP messages:**

1. Open the Network Settings page (**Setup** menu > **IP Network** tab > **Advanced** folder > **Network Settings**).
2. Under the ICMP group, do the following:
   - To enable sending and receipt of ICMP Redirect messages, configure the 'Send and Received ICMP Redirect Messages' [DisableICMPRedirects] parameter to **Enable**.
   - To enable sending of ICMP Destination Unreachable messages, configure the 'Don't Send ICMP Unreachable Messages' [DisableICMPUnreachable] parameter to **Disable**.

3. Click **Apply**.

DNS

If you are using fully qualified domain names (FQDN) instead of IP addresses for some of your device configuration, the domain names need to be resolved into IP addresses by Domain Name System servers. The device provides various ways to do this:

- External, third-party DNS servers:
  - Default DNS servers (see Configuring Default DNS Servers below)
  - DNS servers configured for the associated IP network interface (see Configuring IP Network Interfaces on page 124)
- Device's embedded DNS (and SRV) server:
  - Internal DNS table (see Configuring the Internal DNS Table)
  - Internal SRV table (Configuring the Internal SRV Table)

Configuring Default DNS Servers

The device provides two default DNS server addresses (primary and secondary), which you can modify. The default primary DNS server's address is 8.8.8.8 and the default secondary DNS server's address is 8.8.4.4. The default DNS servers ensure that applications (for example, Automatic Update feature) that may require DNS lookups run seamlessly when you have not configured any DNS servers in the Internal DNS table and IP Interfaces table. In other words, the device uses the default DNS server as the last resort. For example, if you are pinging a destination that is configured as an FQDN, the device searches for a DNS server in the following order:

1. DNS configured for the associated IP interface in the IP Interfaces table (see Configuring IP Network Interfaces on page 124)
2. Default DNS server (this section)

The default DNS servers are used only for certain applications such as Configuration Wizard, CLI ping command, and the Automatic Update feature. It is not used for call routing.

➢ **To modify the default DNS server addresses:**

1. Open the DNS Settings page (**Setup** menu > **IP Network** tab > **DNS** folder > **DNS Settings**).

2. In the 'Default Primary DNS Server IP' field, configure the address of the default primary DNS server.

3. In the 'Default Secondary DNS Server IP' field, configure the address of the default secondary DNS server.
4. Click **Apply**.

Configuring the Internal DNS Table

The Internal DNS table, similar to a DNS resolution can translate up to 20 host (domain) names into IP addresses. This functionality can be used when a domain name (FQDN) is configured as an IP destination in a routing rule. Up to three different IP addresses can be assigned to the same host name.

The device attempts to resolve a domain name into an IP address in the following order:

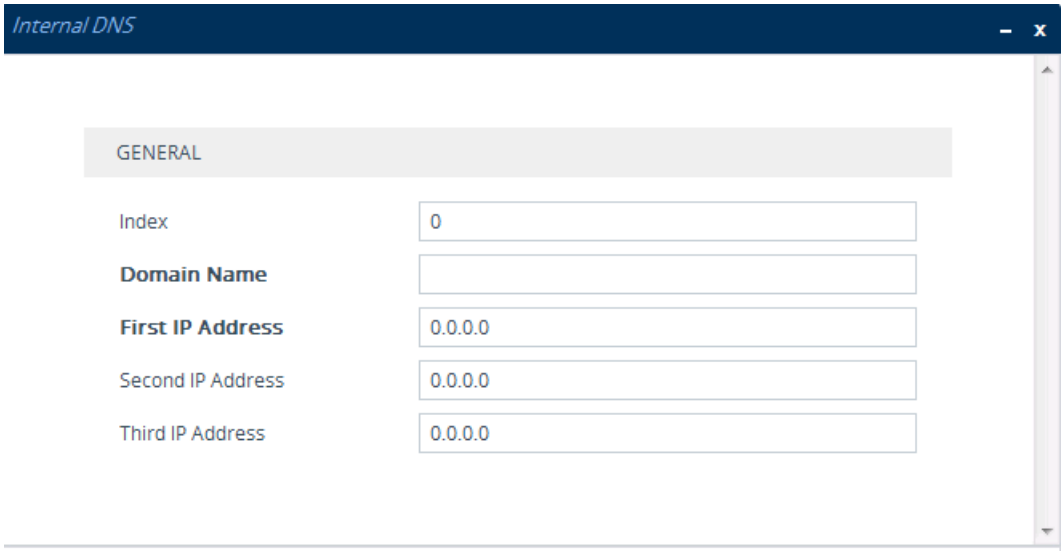
1. The device first checks the Internal DNS table for a matching domain name and if found, resolves the domain name into the corresponding IP address(es).
2. If no matching domain name exists in the Internal DNS table, the device performs a DNS query with an external third-party DNS server whose address is configured for the associated IP network interface in the IP Interfaces table (see Configuring IP Network Interfaces).

The device uses the Internal DNS table only for call routing, for example:
- Call routing according to a SIP Request-URI that contains a hostname.
- Call routing by destination address that is configured as a hostname.
- Call routing by ENUM and the result of the ENUM query is a hostname.
- DNS resolution of proxy servers in a Proxy Set that are configured with an FQDN.
- Registering a user agent whose REGISTER message has a Contact header that is a hostname.

The following procedure describes how to configure the DNS table through the Web interface. You can also configure it through ini file [DNS2IP] or CLI (`configure network > dns dns-to-ip`).

➢ **To configure the device's DNS table:**

1. Open the Internal DNS table (**Setup** menu > **IP Network** tab > **DNS** folder > **Internal DNS**).
2. Click **New**; the following dialog box appears:



3. Configure a DNS rule according to the parameters described in the table below.
4. Click **Apply**.

Table 13-11:Internal DNS Table Parameter Description

Parameter	Description
'Index'	Defines an index number for the new table row. **Note:** Each row must be configured with a unique index.
'Domain Name' `domain-name` [Dns2Ip_ DomainName]	Defines the host name to be translated. The valid value is a string of up to 31 characters.
'First IP Address' `first-ip-address` [Dns2Ip_ FirstIpAddress]	Defines the first IP address (in dotted-decimal format notation) to which the host name is translated. The IP address can be configured as an IPv4 and/or IPv6 address.
'Second IP Address' `second-ip-address` [Dns2Ip_ SecondIpAddress]	Defines the second IP address (in dotted-decimal format notation) to which the host name is translated.
'Third IP Address' `third-ip-address` [Dns2Ip_	Defines the third IP address (in dotted-decimal format notation) to which the host name is translated.

Parameter	Description
ThirdIpAddress]	

Configuring the Internal SRV Table

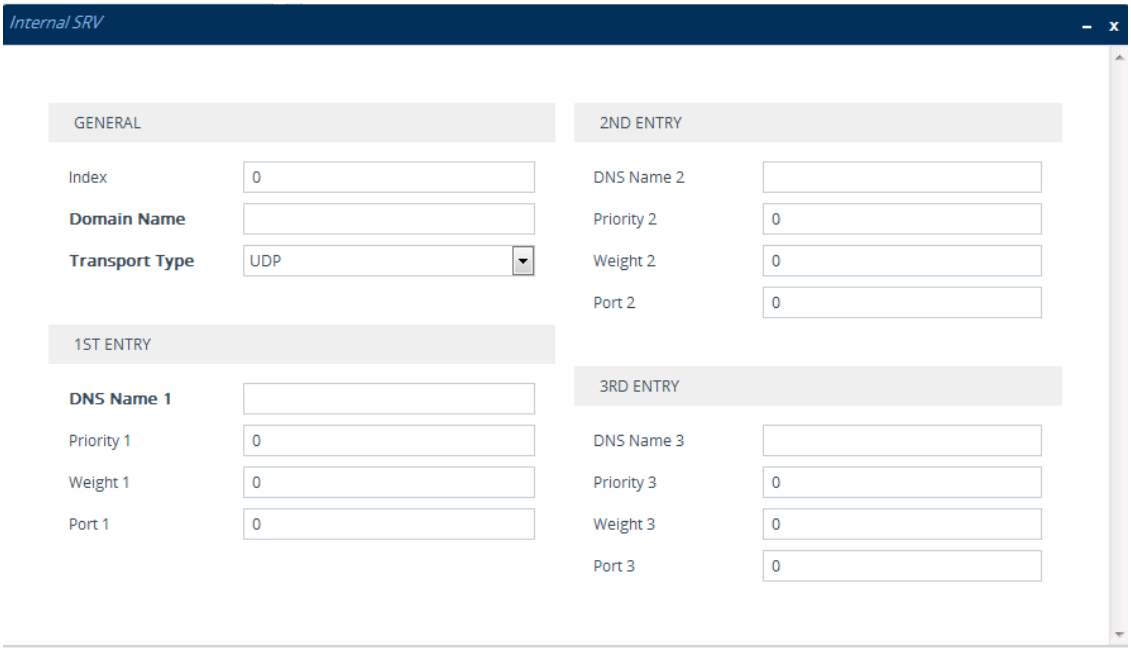
The Internal SRV table lets you configure up to 10 SRV rows. The table is used to resolve hostnames into DNS A-Records. You can assign three different A-Records per hostname, where each A-Record includes the hostname, priority, weight, and port.

The device first attempts to resolve a domain name using this table. If the domain name is not configured in this table, the device performs an SRV resolution using an external DNS server whose address is configured for the associated IP network interface in the IP Interfaces table. If the IP network interface is not configured with a DNS address, the device performs an SRV resolution with the default external DNS server (see Configuring Default DNS Servers on page 152).

The following procedure describes how to configure the Internal SRV table through the Web interface. You can also configure it through ini file [SRV2IP] or CLI (`configure network > dns srv2ip`).

➢ To configure the device's SRV table:

1. Open the Internal SRV table (**Setup** menu > **IP Network** tab > **DNS** folder > **Internal SRV**).
2. Click **New**; the following dialog box appears:



3. Configure an SRV rule according to the parameters described in the table below.
4. Click **Apply**, and then save your settings to flash memory.

Table 13-12:Internal SRV Table Parameter Descriptions

Parameter	Description
General	
'Index'	Defines an index number for the new table row. **Note:** Each row must be configured with a unique index.
'Domain Name' `domain-name` [Srv2Ip_ InternalDomain]	Defines the hostname to be translated. The valid value is a string of up to 31 characters. By default, no value is defined.
'Transport Type' `transport-type` [Srv2Ip_ TransportType]	Defines the transport type. ■ [0] **UDP** (default) ■ [1] **TCP** ■ [2] **TLS**
1st/2nd/3rd Entry	
'DNS Name (1-3)' `dns-name-1|2|3` [Srv2Ip_Dns1/2/3]	Defines the first, second or third DNS A-Record to which the hostname is translated. By default, no value is defined.
'Priority (1-3)' `priority-1|2|3` [Srv2Ip_ Priority1/2/3]	Defines the priority of the target host. A lower value means that it is more preferred. By default, no value is defined.
'Weight (1-3)' `weight-1|2|3` [Srv2Ip_ Weight1/2/3]	Defines a relative weight for records with the same priority. By default, no value is defined.
'Port (1-3)' `port-1|2|3` [Srv2Ip_Port1/2/3]	Defines the TCP or UDP port on which the service is to be found. By default, no value is defined.

IP Multicasting

The device supports IP Multicasting level 1, according to RFC 2236 (i.e., IGMP version 2) for RTP channels. The device is capable of transmitting and receiving multicast packets.

14 Security

This section describes the VoIP security-related configuration.

Configuring TLS Certificates

The TLS Contexts table lets you configure X.509 certificates which are used for secure management of the device, secure SIP transactions, and other security applications.

- The device is shipped with an active, default TLS setup (TLS Context ID 0, named "default"). Therefore, configure certificates only if required.
- Since X.509 certificates have an expiration date and time, you must configure the device to use Network Time Protocol (NTP) to obtain the current date and time from an NTP server. Without the correct date and time, client certificates cannot work. To configure NTP, see Configuring Automatic Date and Time using SNTP.
- Only **Base64 (PEM)** encoded X.509 certificates can be loaded to the device.

Configuring TLS Certificate Contexts

The TLS Contexts table lets you configure up to 100 TLS Contexts. A TLS Context defines Transport Layer Security (TLS) settings (e.g., TLS certificates). The TLS protocol provides confidentiality, integrity, and authenticity between two communicating applications over TCP/IP. You can use TLS for the following:

- To secure device management communication, for example, HTTPS-based Web sessions, Telnet sessions and SSH sessions.
- To secure SIP signaling connections, referred to as SIP Secure (SIPS) or SIP over TLS.
- To secure various other network applications supported by the device, for example, communication with a remote LDAP server used for LDAP-based user management authentication and authorization.

The device is shipped with a default TLS Context (Index #0 and named "default"), which includes a self-generated random private key and a self-signed server certificate. The Common Name (CN or subject name) of the default certificate is "ACL_nnnnnnn", where *nnnnnnn* denotes the serial number of the device. If the default self-signed certificate is about to expire (less than a day), the device automatically re-generates a new self-signed certificate.

- The default TLS Context cannot be deleted.
- For secure management through the default management network interface (i.e., **OAMP** Application Type in the IP Interfaces table), the device uses the default TLS Context. However, for secure Web- and REST-based management through Additional Management Interfaces (configured in Configuring Additional Management Interfaces on page 46), you can use any TLS Context.
- If a TLS Context for an existing TLS connection is changed during the call by the user agent, the device ends the connection.
- For more information on secured management, see Configuring Secured (HTTPS) Web on page 64.

You can configure each TLS Context with the following TLS settings:

- TLS version (TLS 1.0, TLS 1.1, TLS 1.2, and TLS 1.3).
- DTLS version (DTLS 1.0 and DTLS 1.2).
- TLS cipher suites for server and client roles (per OpenSSL syntax).
- Diffie-Hellman (DH) key size used by the device if it acts as a TLS server and DH is used for key exchange.
- TLS certificate expiry check, whereby the device periodically checks the validation date of the installed TLS server certificates and sends an SNMP trap event if a certificate is nearing expiry. To configure TLS certificate expiry check, see Configuring TLS Server Certificate Expiry Check.
- Online Certificate Status Protocol (OCSP). Some Public-Key Infrastructures (PKI) can revoke a certificate after it has been issued. You can configure the device to check if a peer's certificate has been revoked, using the OCSP. When OCSP is enabled, the device queries the OCSP server for revocation information whenever a peer certificate is received (TLS client mode, or TLS server mode with mutual authentication).

- The device does not query OCSP for its own certificate.
- Some PKIs do not support OCSP, but generate Certificate Revocation Lists (CRLs). For such scenarios, set up an OCSP server such as OCSPD.

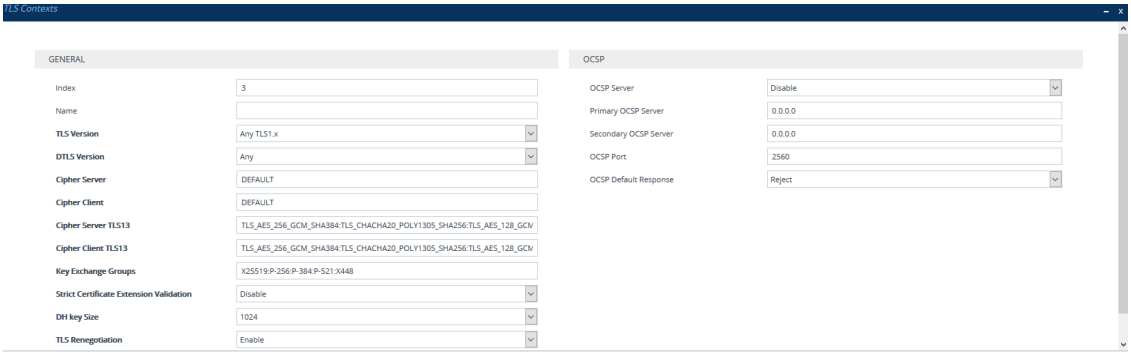
- Private key - externally created and then uploaded to device.
- Different levels of security strength (key size) per TLS certificate.
- X.509 certificates - self-signed certificates or signed as a result of a certificate signing request (CSR).
- Trusted root certificate authority (CA) store (for validating certificates).

- When creating a TLS Context, you should create a certificate as described in Creating Self-Signed Certificates for TLS Contexts on page 172, and then check that the certificate is "Ok" as described in Viewing Certificate Information on page 168.
- For secure SIP messaging (SIP Secure or SIPS) using TLS, see TLS for SIP Clients on page 177 (two-way authentication) and Configuring TLS for Secured SIP on page 176.

The following procedure describes how to configure a TLS Context through the Web interface. You can also configure it through ini file [TLSContexts] or CLI (`configure network > tls`).

➢ To configure a TLS Context:

1. Open the TLS Contexts table (**Setup** menu > **IP Network** tab > **Security** folder > **TLS Contexts**).
2. Click **New** to add a new TLS Context or **Edit** to modify the default TLS Context at Index 0; the following dialog box appears:



3. Configure the TLS Context according to the parameters described in the table below.
4. Click **Apply**, and then reset the device with a save-to-flash for your settings to take effect.

Table 14-1: TLS Contexts Parameter Descriptions

Parameter	Description
General	
'Index' `tls` [TLSContexts_Index]	Defines an index number for the new table row. **Note:** ■ Each row must be configured with a unique index. ■ Index 0 ("default") is the default TLS Context.
'Name' `name`	Defines a descriptive name, which is used when associating the row in other tables.

Parameter	Description
[TLSContexts_Name]	The valid value is a string of up to 31 characters. **Note:** ■ The parameter value cannot contain a forward slash (/). ■ The default TLS Context (Index 0) is named "default".
'TLS Version' `tls-version` [TLSContexts_TLSVersion]	Defines the supported TLS protocol version. Clients attempting to communicate with the device using a different TLS version are rejected. ■ [0] **Any TLS1.x** = (Default) TLSv1.0, TLSv1.1, TLSv1.2, and TLSv1.3 are supported. ■ [1] **TLSv1.0** = Only TLS 1.0. ■ [2] **TLSv1.1** = Only TLS 1.1. ■ [3] **TLSv1.0 and TLSv1.1** = Only TLS 1.0 and TLS 1.1. ■ [4] **TLSv1.2** = Only TLS 1.2. ■ [6] **TLSv1.1 and TLSv1.2** = Only TLS 1.1 and TLS 1.2. ■ [7] **TLSv1.0 TLSv1.1 and TLSv1.2** = Only TLS 1.0, TLS 1.1, and TLS 1.2. ■ [8] **TLSv1.3** = Only TLS 1.3. ■ [12] **TLSv1.2 and TLSv1.3** = Only TLS 1.2 and 1.3. ■ [14] **TLSv1.1 TLSv1.2 and TLSv1.3** = Only TLS 1.1, TLS 1.2, and TLS 1.3. ■ [15] **TLSv1.0 TLSv1.1 TLSv1.2 and TLSv1.3** = Only TLS 1.0, TLS 1.1, TLS 1.2, and TLS 1.3.
'DTLS Version' [TLSContexts_DTLSVersion]	Defines the Datagram Transport Layer Security (DTLS) version, which is used to negotiate keys for WebRTC calls. ■ [0] **Any** (default) ■ [1] **DTLSv1.0** ■ [2] **DTLSv1.2** For more information on WebRTC, see WebRTC.
'Cipher Server' `ciphers-server` [TLSContexts_	Defines the supported cipher suite for the TLS server (in OpenSSL cipher list format) when the TLS version is 1.2 or earlier.

Parameter	Description
ServerCipherString]	For possible values and additional details, visit the OpenSSL website. The default is "DEFAULT". **Note:** The parameter is applicable only to TLS 1.2 and earlier.
'Cipher Client' `ciphers-client` [TLSContexts_ ClientCipherString]	Defines the supported cipher suite for TLS clients when the TLS version is 1.2 or earlier. For possible values and additional details, visit the OpenSSL website. The default is "DEFAULT". **Note:** The parameter is applicable only to TLS 1.2 and earlier.
'Cipher Server TLS1.3' `ciphers-server-tls13` [TLSContexts_ ServerCipherTLS13String]	Defines the supported cipher suite for the TLS 1.3 server (in OpenSSL cipher list format). For possible values and additional details, visit the OpenSSL website. The default is "TLS_AES_256_GCM_ SHA384:TLS_CHACHA20_POLY1305_SHA256:TLS_AES_ 128_GCM_SHA256". **Note:** The parameter is applicable only to TLS 1.3.
'Cipher Client TLS1.3' `ciphers-client-tls13` [TLSContexts_ ClientCipherTLS13String]	Defines the supported cipher suite for TLS 1.3 clients. For possible values and additional details, visit the OpenSSL website. The default is "TLS_AES_256_GCM_ SHA384:TLS_CHACHA20_POLY1305_SHA256:TLS_AES_ 128_GCM_SHA256". **Note:** The parameter is applicable only to TLS 1.3.
'Key Exchange Groups' `key-exchange-groups` [TLSContexts_ KeyExchangeGroups]	Defines the groups that are supported for key exchange, ordered from most preferred to least preferred. The valid value is any combination of the following strings: ■ X25519 ■ P-256 ■ P-384 ■ P-521 ■ X448 The default is "X25519:P-256:P-384:X448" (without quotation marks). When configuring the parameter with multiple values, separate each with a colon. In addition, the order of the

Parameter	Description
	values determines the group preference. For example, the value "P-384:P-256:X25519" (without quotation marks) gives preference to P-384. The TLS client uses the first configured value (e.g., P-384) as its group trial, while the TLS server uses the whole list to try and match the client's trial. **Note:** The parameter is applicable to all TLS versions.
'Strict Certificate Extension Validation' `require-strict-cert` [TLSContexts_ RequireStrictCert]	Enables the validation of the extensions (keyUsage and extendedKeyUsage) of peer certificates. The validation ensures that the signing CA is authorized to sign certificates and that the end-entity certificate is authorized to negotiate a secure TLS connection. ■ [0] **Disable** (default) ■ [1] **Enable**
'DH Key Size' `dh-key-size` [TLSContexts_DHKeySize]	Defines the Diffie-Hellman (DH) key size (in bits). DH is an algorithm used mainly for exchanging cryptography keys used in symmetric encryption algorithms such as AES. ■ [1024] **1024 - Not Recommended** ■ [2048] **2048** (default) ■ [3072] **3072** **Note:** 1024-bit key size is not recommended.
'TLS Renegotiation' `tls-renegotiation` [TLSContexts_ TlsRenegotiation]	Enables TLS renegotiations (handshakes) initiated by the client (peer) with the device. ■ [0] **Disable** = The device blocks client-initiated TLS renegotiations and allows only one TLS handshake process. This is useful, for example, for preventing Denial-of-Service (DoS) attacks on the device caused by multiple TLS renegotiations per second by an attacker. ■ [1] **Enable** (default)
OCSP	
'OCSP Server' `ocsp-server` [TLSContexts_OcspEnable]	Enables or disables certificate checking using OCSP. ■ [0] **Disable** (default) ■ [1] **Enable**

Parameter	Description
'Primary OCSP Server' ocsp-server-primary [TLSContexts_ OcspServerPrimary]	Defines the IP address (in dotted-decimal notation) of the primary OCSP server. The default is 0.0.0.0.
'Secondary OCSP Server' ocsp-server-secondary [TLSContexts_ OcspServerSecondary]	Defines the IP address (in dotted-decimal notation) of the secondary OCSP server (optional). The default is 0.0.0.0.
'OCSP Port' ocsp-port [TLSContexts_ OcspServerPort]	Defines the OCSP server's TCP port number. The default port is 2560.
'OCSP Default Response' ocsp-default-response [TLSContexts_ OcspDefaultResponse]	Determines whether the device allows or rejects peer certificates if it cannot connect to the OCSP server. ■ [0] **Reject** (default) ■ [1] **Allow**

Assigning CSR-based Certificates to TLS Contexts

You can request a digitally signed certificate from a Certification Authority (CA) for a TLS Context. This process is referred to as a certificate signing request (CSR) and is required if your organization employs a Public Key Infrastructure (PKI) system. The CSR contains information identifying the device such as a Distinguished Name (DN) or subject alternative names in the case of an X.509 certificate.

➢ **To assign a CSR-based certificate to a TLS Context:**

1. Open the TLS Contexts table (see Configuring TLS Certificate Contexts).
2. In the table, select the required TLS Context, and then click the **Change Certificate** link located below the table; the Change Certificates page appears.
3. Under the **Certificate Signing Request** group, fill in the following information:
   a. Distinguished Name (DN) fields (uniquely identifies the device):
      - In the 'Common Name [CN]' field, enter the common name.
      - (Optional) In the 'Organizational Unit [OU]' field, enter the section of the organization.

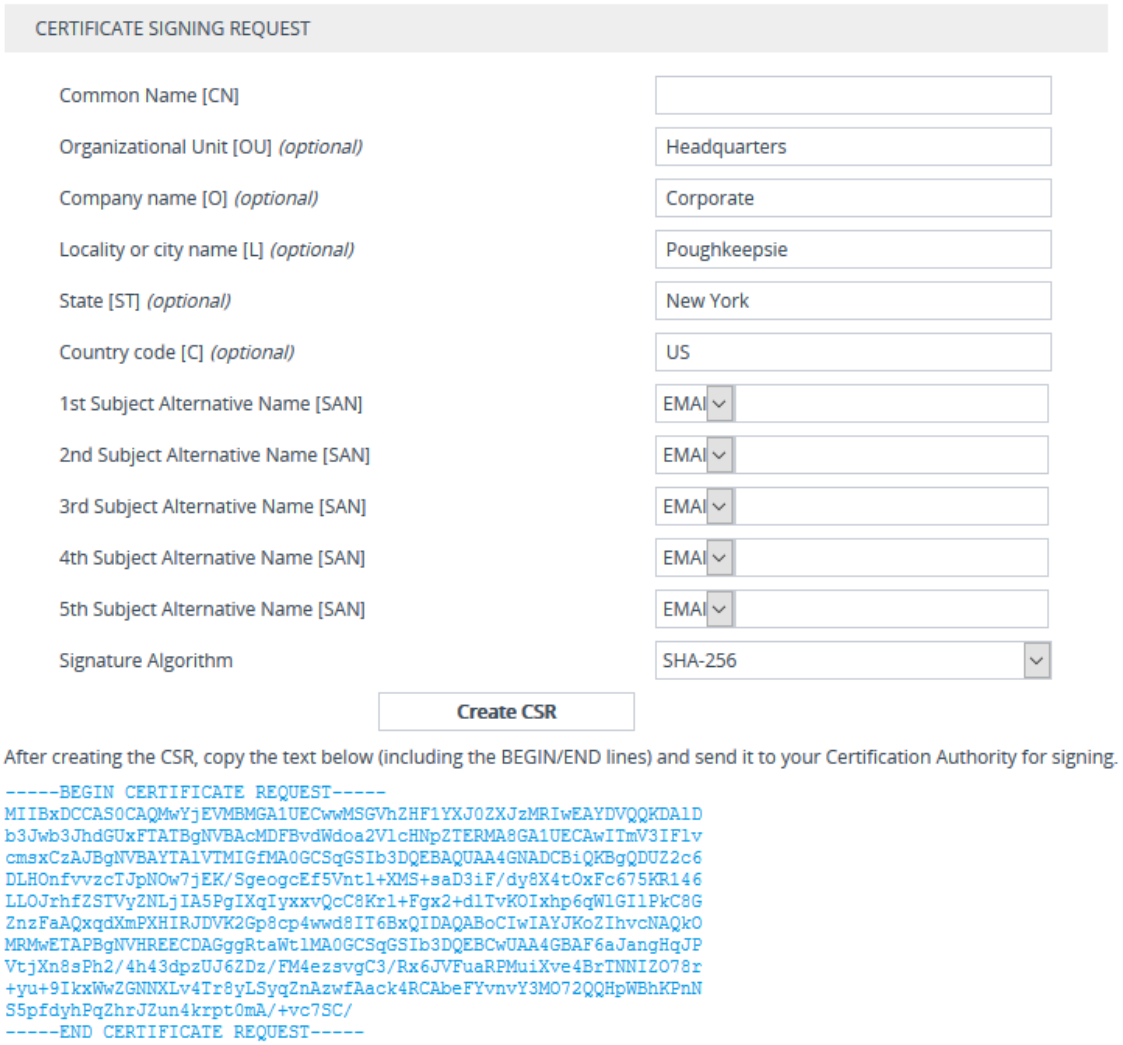
- (Optional) In the ' Company name [O]' field, enter the legal name of your organization.
- (Optional) In the 'Locality or city name [L]' field, enter the city where your organization is located.
- (Optional) In the ' State [ST]' field, enter the state or province where your organization is located.
- (Optional) In the ' Country code [C]' field, enter the two-letter ISO abbreviation for your country.

b. If you want to generate a CSR for SAN (with multiple subject alternate names), then from the 'Subject Alternative Name [SAN]' drop-down list, select the type of SAN (e-mail address, DNS hostname, URI, or IP address), and then enter the relevant value. You can configure multiple SAN names, using the 1st to 5th 'Subject Alternative Name [SAN]' fields.

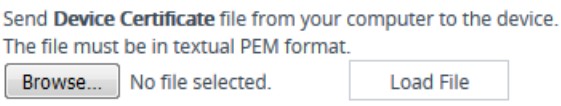
c. From the 'Signature Algorithm' drop-down list, select the hash function algorithm (SHA-1, SHA-256, or SHA-512) with which to sign the certificate.

- Fill in the fields according to your security provider's instructions.
- If you leave the 'Common Name [CN]' field empty, the device generates the CSR with the default Common Name (**CN=ACL_<6-digit serial number of device>**).

4. Click the **Create CSR** button; a textual certificate signing request is displayed in the area below the button:

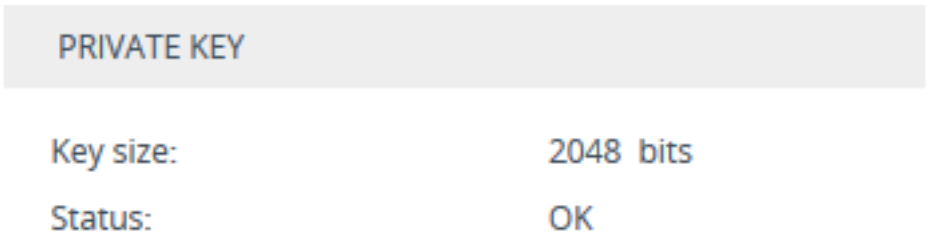


5. Copy the text and send it to your security provider (CA) to sign this request.
6. When the CA sends you a server certificate, save the certificate to a file (e.g., cert.txt). Make sure that the file is a plain-text file containing the "BEGIN CERTIFICATE" header.
7. Scroll down to the **Upload Certificates Files From Your Computer** group, click the **Browse** button corresponding to the 'Send Device Certificate...' field, navigate to the cert.txt file, and then click **Load File**:



8. Wait for the certificate to successfully load to the device.
9. Save configuration with a device reset.
10. Verify that the private key is correct:
    a. Open the TLS Contexts table, and then select the TLS Context.
    b. Click the **Certificate Information** link located below the table.

c. Make sure that the 'Status' field displays "OK"; otherwise, consult with your security administrator:



- The certificate replacement process can be repeated whenever necessary (e.g., when the new certificate expires).
- You can also load the certificate through the device's Automatic Provisioning mechanism, using the [HTTPSCertFileName] parameter.

TLS Context Parameters Relevancy per Application

The following table shows the parameters of the TLS Context table that are used when establishing a TLS connection per SBC application.

Application	TLS Contexts Table Parameter								
	'TLS Version'	'Cipher Server'	'Cipher Client'	'Cipher Server TLS1.3'	'Cipher Client TLS1.3'	'Key Exchange Groups'	'Strict Certificate Extension Validation'	'DH key Size'	'TLS Renegotiation'
Web / REST Server	✓	✓	✓	✓	✓	✓	✓	✓	✓
Automatic Update	✓	-	✓	-	✓	-	-	-	-
CLI `copy` Com-mands	✓	-	✓	-	✓	-	-	-	-
Sending CDRs to Remote Server	✓	-	✓	-	✓	-	-	-	-

Application	TLS Contexts Table Parameter								
	'TLS Version'	'Cipher Server'	'Cipher Client'	'Cipher Server TLS1.3'	'Cipher Client TLS1.3'	'Key Exchange Groups'	'Strict Certificate Extension Validation'	'DH key Size'	'TLS Renegotiation'
HTTPS Proxy	✓	✓	✓	-	-	-	-	-	-
Secure Communication with OVOC	✓	✓	✓	✓	✓	✓	✓	✓	✓
WebSocket Tunnel with OVOC	✓	-	✓	-	✓	✓	✓	-	-
Secured LDAP Client	✓	✓	✓	✓	✓	✓	✓	✓	✓
Secured SCTP	✓	✓	✓	✓	✓	✓	✓	✓	✓
TR-069	✓	✓	✓	✓	✓	✓	✓	✓	✓
ZeroConf Provisioning	✓	✓	✓	✓	✓	✓	✓	✓	✓

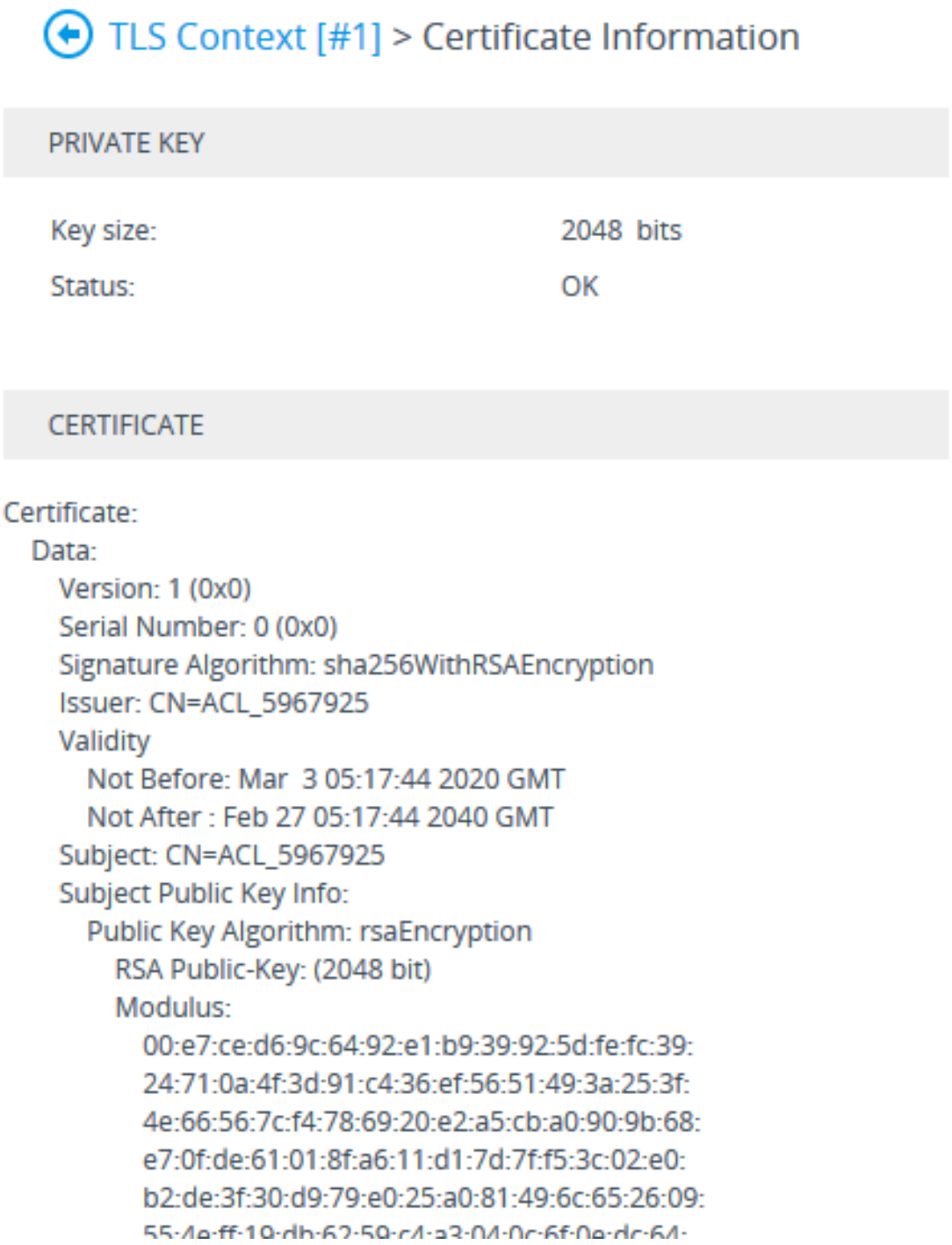
Viewing Certificate Information

You can view information of TLS certificates installed on the device per TLS Context.

➢ **To view certificate information:**

1. Open the TLS Contexts table (see Configuring TLS Certificate Contexts).

2. Select a TLS Context, and then click the **Certificate Information** link located below the table; the Certificate Information page appears, showing the certificate information. The following figure shows an example (but cropped due to space).



Assigning Externally Created Private Keys to TLS Contexts

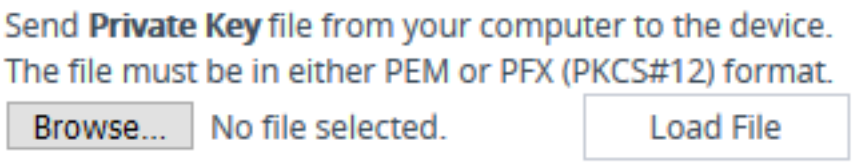
The following procedure describes how to assign an externally created private key to a TLS Context.

➢ **To assign an externally created private key to a TLS Context:**

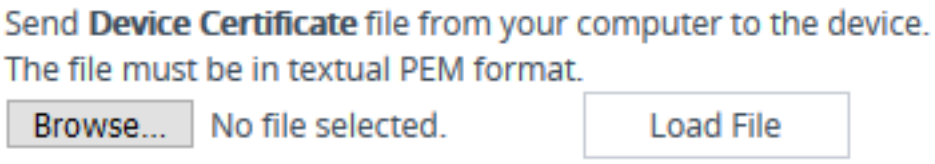
1. Obtain a private key in either textual PEM (PKCS #7) or PFX (PKCS #12) format (typically provided by your security administrator). The file may be encrypted with a short pass-phrase.
2. Open the TLS Contexts table (see Configuring TLS Certificate Contexts).
3. In the table, select the required TLS Context, and then click the **Change Certificate** link located below the table; the Change Certificates page appears.
4. Scroll down to the **Upload Certificate Files From Your Computer** group.
   a. (Optional) In the 'Private key pass-phrase' field, enter the password (passphrase) of the encrypted private key file. The default passphrase is "audc". The passphrase can be up to 32 characters. If there is no passphrase, leave the field blank.

The passphrase cannot be configured with wide characters.

   b. Load the private key file (see Step 1), by selecting it using the **Browse** button corresponding to the '**Send Private** Key file ...' text, and then clicking **Load File**.



   c. If your security administrator has provided you with a device certificate file, load it by selecting the file using the **Browse** button corresponding to the '**Send Device** Certificate file ...' text, and then clicking **Load File**.



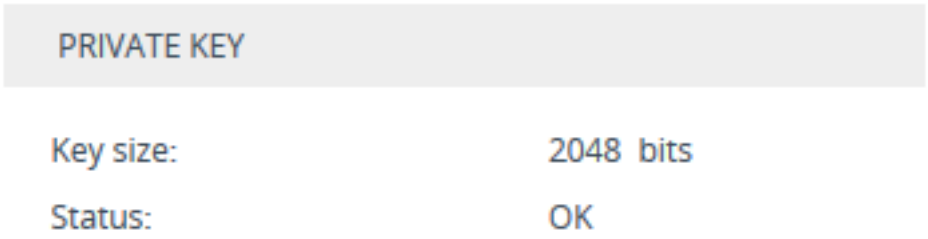
The loaded private key file must match the loaded device certificate file.

5. After the files successfully load to the device, save the configuration with a device reset.
6. Verify that the private key is correct:
   a. Open the TLS Contexts table.

b. Select the required TLS Context index row.

c. Click the **Certificate Information** link located below the table.

d. Make sure that the 'Status' field displays "OK"; otherwise (i.e., displays "Does not match certificate"), consult with your security administrator:

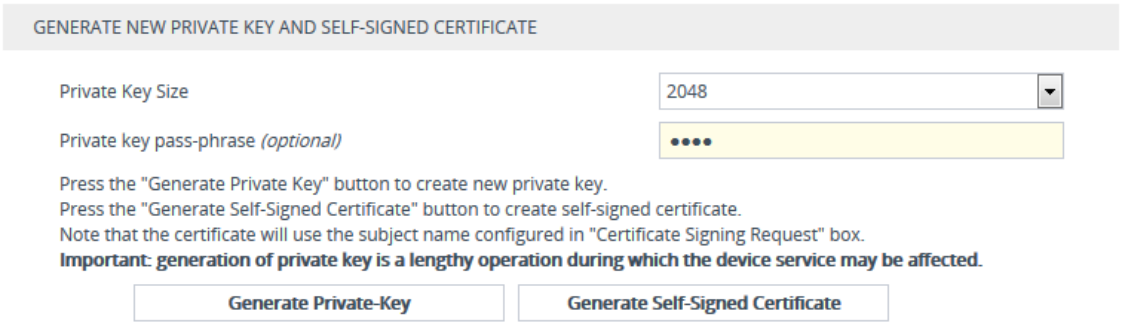


Generating Private Keys for TLS Contexts

The device can generate the private key for a TLS Context. The private key can be generated for CSR or self-signed certificates.

➢ **To generate a new private key for a TLS Context:**

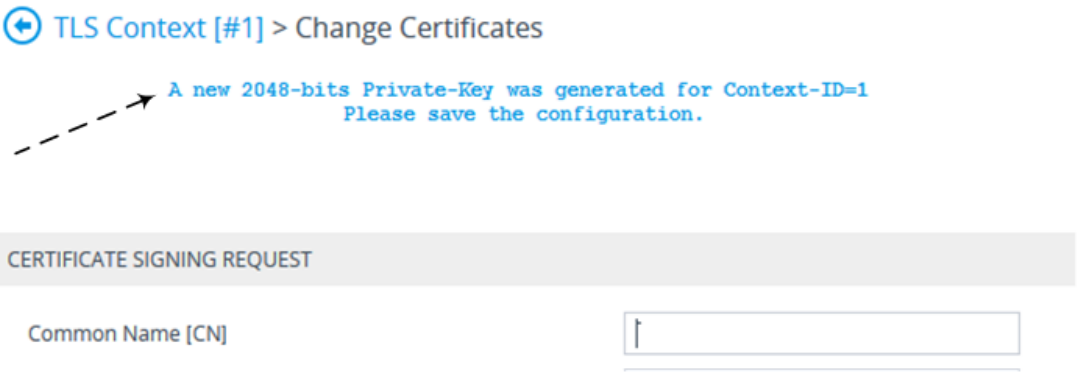
1. Open the TLS Contexts table (see Configuring TLS Certificate Contexts).
2. In the table, select the required TLS Context index row, and then click the **Change Certificates** link located below the table; the Change Certificates page appears.
3. Scroll down to the **Generate New Private Key and Self-signed Certificate** group:



4. From the 'Private Key Size' drop-down list, select the private key size (in bits) for RSA public-key encryption for newly self-signed generated keys:
   - **1024 - Not Recommended**
   - **2048** (default)
   - **3072**
   - **4096**
5. (Optional) In the 'Private key pass-phrase' field, enter a password (passphrase) to encrypt the private key file. The default passphrase is "audc". The passphrase can be up to 32 characters. If you don't want to encrypt the file, leave the field blank.

> The passphrase cannot be configured with wide characters.

6. Click **Generate Private-Key**; a message appears requesting you to confirm key generation.
7. Click **OK** to confirm key generation; the device generates a new private key, indicated by a message in the **Certificate Signing Request** group:



8. Continue with certificate configuration by creating a CSR or generating a new self-signed certificate.
9. Save configuration with a device reset for the new certificate to take effect.

Creating Self-Signed Certificates for TLS Contexts

You can assign a certificate that is digitally signed by the device itself to a TLS Context (i.e., self-signed certificate). In other words, the device acts as a CA. The Issuer (e.g., "Issuer: CN=ACL_5967925") and Subject (e.g., " Subject: CN=ACL_5967925") fields of the self-signed certificate have the same value.

- The device is shipped with a default TLS Context (Index 0 and named "default"), which includes a self-generated random private key and a self-signed server certificate. The Common Name (CN or subject name) of the default certificate is "ACL_nnnnnnn", where *nnnnnnn* denotes the serial number of the device.
- If the default self-signed certificate is about to expire (less than a day) or has expired, the device automatically re-generates a new self-signed certificate.

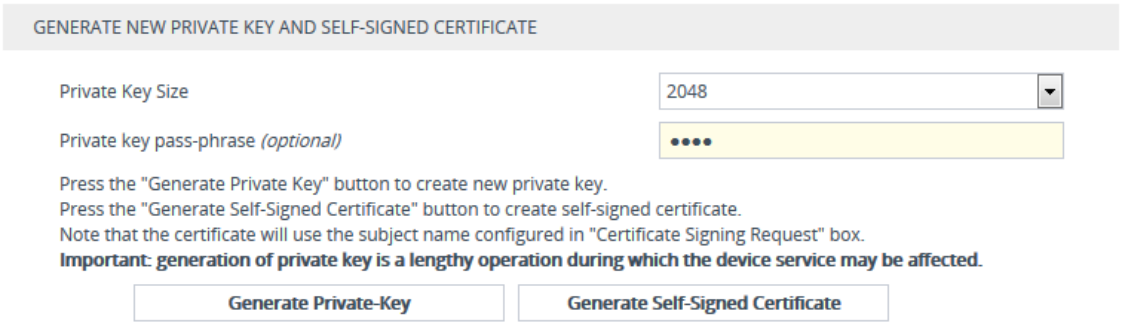
You can configure each TLS Context with the following:

➢ To assign a self-signed certificate to a TLS Context:

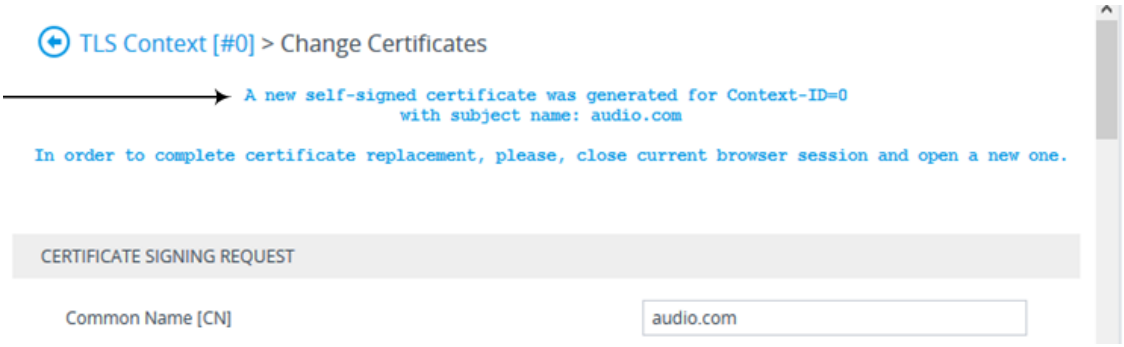
1. Before you begin, make sure of the following:
   - You have a unique DNS name for the device (e.g., dns_name.corp.customer.com). The name is used to access the device and therefore, must be listed in the server certificate.

- No traffic is running on the device. The certificate generation process is disruptive to traffic and should be done during maintenance time.

2. Open the TLS Contexts table (see Configuring TLS Certificate Contexts).
3. In the table, select the required TLS Context index row, and then click the **Change Certificate** link located below the table; the Change Certificates page appears.
4. Under the **Certificate Signing Request** group, in the 'Common Name [CN]' field, enter the fully-qualified DNS name (FQDN) as the certificate subject. Alternatively (or in addition), if you want to generate a self-signed SAN certificate (with multiple subject alternate names), then from the 'Subject Alternative Name [SAN]' drop-down list, select the type of SAN (e-mail address, DNS hostname, URI, or IP address), and then enter the relevant value. You can configure multiple SANs, using the 1st to 5th 'Subject Alternative Name [SAN]' fields.
5. Scroll down the page to the **Generate New Private Key and Self-signed Certificate** group:



6. Click **Generate Self-Signed Certificate**; a message appears requesting you to confirm generation.
7. Click **OK** to confirm generation; the device generates a new self-signed certificate displaying the new subject name, indicated by a message in the **Certificate Signing Request** group:

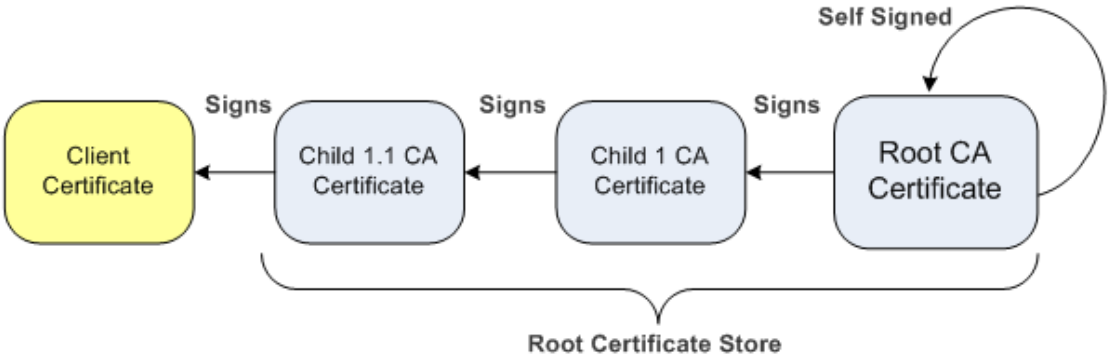


8. Save the configuration with a device reset for the new certificate to take effect.

Importing Certificates into Trusted Root Certificate Store

The device provides its own Trusted Root Certificate store. This lets you manage certificate trust. You can import up to 20 certificates to the store per TLS Context (but this may be less depending on certificate file size).

The store can also be used for certificate chains. A certificate chain is a sequence of certificates where each certificate in the chain is signed by the subsequent certificate. The last certificate in the list of certificates is the Root CA certificate, which is self-signed. The purpose of a certificate chain is to establish a chain of trust from a child certificate to the trusted root CA certificate. The CA vouches for the identity of the child certificate by signing it. A client certificate is considered trusted if one of the CA certificates up the certificate chain is found in the server certificate directory. For the device to trust a whole chain of certificates per TLS Context, you need to import them into the device's Trusted Certificates Store, as described below.

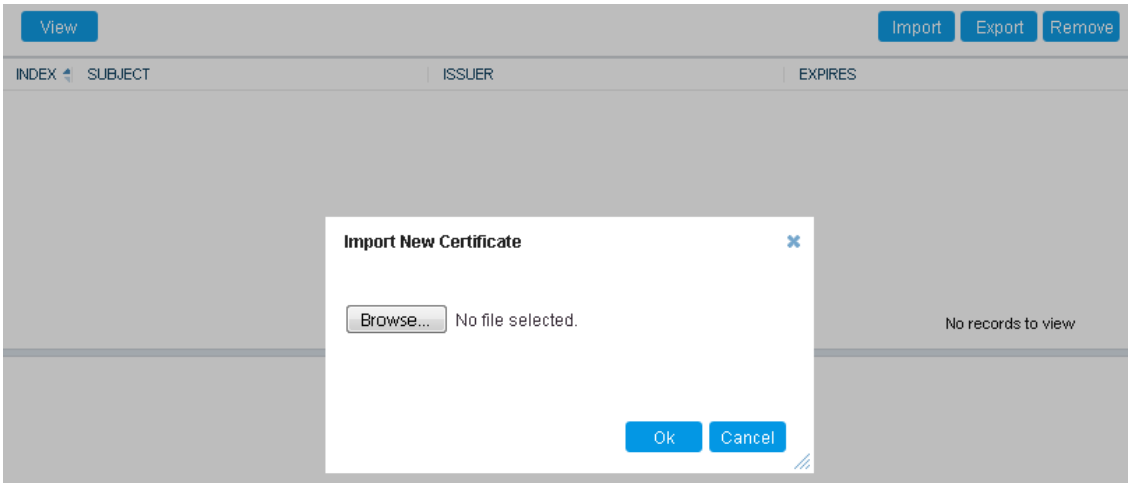


You can also import multiple TLS root certificates in bulk from a single file. Each certificate in the file must be Base64 encoded (PEM). When copying-and-pasting the certificates into the file, each Base64 ASCII encoded certificate string must be enclosed between "**-----BEGIN CERTIFICATE-----**" and "**-----END CERTIFICATE-----**".

> Only Base64 (PEM) encoded X.509 certificates can be loaded to the device.

➢ **To import certificates into the Trusted Root Certificate Store:**

1. Open the TLS Contexts table (see Configuring TLS Certificate Contexts).
2. In the table, select the required TLS Context, and then click the **Trusted Root Certificates** link located below the table; the Trusted Certificates table appears.
3. Click the **Import** button, and then browse to and select the certificate file.



4. Click **OK**; the certificate is loaded to the device and listed in the Trusted Certificates store.

You can also do the following with certificates that are in the Trusted Certificates store:

- Delete certificates: Select the required certificate, click **Remove**, and then in the Remove Certificate dialog box, click **Remove**.
- Save certificates to a folder on your PC: Select the required certificate, click **Export**, and then in the Export Certificate dialog box, browse to the folder on your PC where you want to save the file and click **Export**.

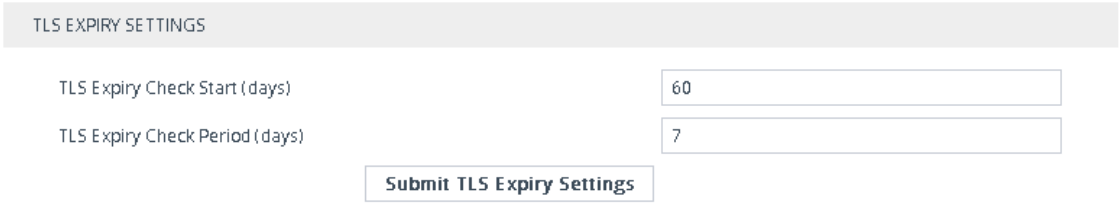
Configuring TLS Server Certificate Expiry Check

You can configure the TLS Server Certificate Expiry Check feature per TLS Context, whereby the device periodically checks the validation date of installed TLS server certificates. You can also configure the device to send an SNMP alarm (acCertificateExpiryAlarm) at a user-defined number of days before the installed TLS server certificate is to expire. The alarm indicates the TLS Context to which the certificate belongs.

> **Note:** If the device's default self-signed certificate (at TLS Context Index 0 and named "default") is about to expire (less than a day) or has expired, the device automatically re-generates a new self-signed certificate. The configuration described in this section does not apply to this mechanism (occurs regardless).

➢ **To configure TLS certificate expiry checks and notification:**

1. Open the TLS Contexts table (see Configuring TLS Certificate Contexts).
2. Select the required TLS Context index row, and then click the **Change Certificate** link located below the table; the Change Certificates page appears.
3. Scroll down the page to the **TLS Expiry Settings** group:



4. In the 'TLS Expiry Check Start' field, enter the number of days before the installed TLS server certificate is to expire when the device sends an SNMP trap event to notify of this.
5. In the 'TLS Expiry Check Period' field, enter the periodical interval (in days) for checking the TLS server certificate expiry date. By default, the device checks the certificate every 7 days.
6. Click the **Submit TLS Expiry Settings** button.

Configuring TLS for Secured SIP

The device uses TLS over TCP to encrypt and optionally, authenticate SIP messages. This is referred to as SIP Secure (SIPS). SIPS uses the X.509 certificate exchange process. For configuring TLS (TLS Context), see Configuring TLS Certificates on page 158.

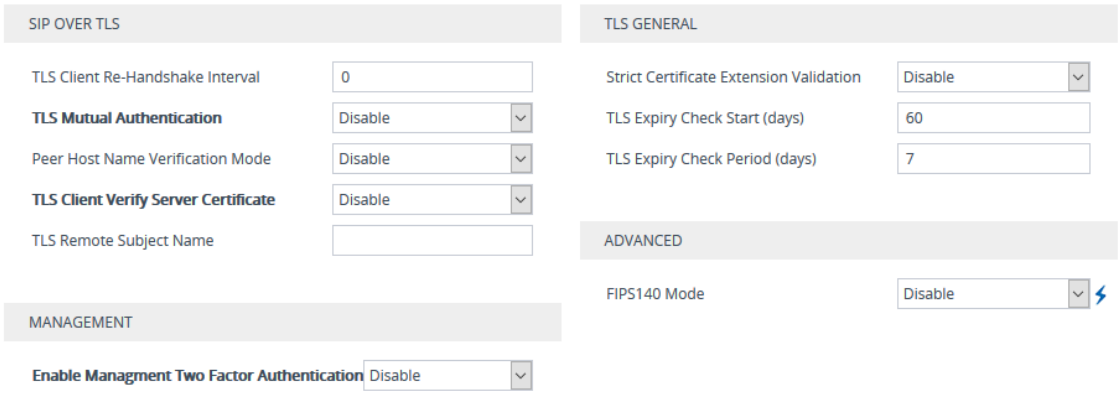
To use a TLS Context for SIPS, you need to assign it to a Proxy Set or SIP Interface (or both) that is associated with the IP Group for which you want to employ TLS. When the device establishes a TLS connection (handshake) with a SIP user agent (UA), the TLS Context is determined as follows:

- **Incoming calls:**
  a. Proxy Set: If the incoming call is successfully classified to an IP Group based on Proxy Set (i.e., IP address of calling party) and the Proxy Set is configured for TLS ('Transport Type' parameter is set to **TLS**), the TLS Context assigned to the Proxy Set is used. To configure Proxy Sets, see Configuring Proxy Sets.
  b. SIP Interface: If the Proxy Set is either not configured for TLS (i.e., the 'Transport Type' parameter is set to **UDP**) or not assigned a TLS Context, and/or classification to a Proxy Set fails, the device uses the TLS Context assigned to the SIP Interface used for the call. To configure SIP Interfaces, see Configuring SIP Interfaces.
  c. Default TLS Context (Index #0): If the SIP Interface is not assigned a TLS Context or no SIP Interface is used for the call, the device uses the default TLS Context.
- **Outgoing calls:**
  a. Proxy Set: If the outgoing call is sent to an IP Group associated with a Proxy Set that is assigned a TLS Context and the Proxy Set is configured for TLS (i.e., 'Transport Type' parameter is set to **TLS**), the TLS Context is used. If the 'Transport Type' parameter is set to **UDP**, the device uses UDP to communicate with the proxy and no TLS Context is used.
  b. SIP Interface: If the Proxy Set is not assigned a TLS Context, the device uses the TLS Context assigned to the SIP Interface used for the call.
  c. Default TLS Context (Index #0): If the SIP Interface is not assigned a TLS Context or no SIP Interface is used for the call, the device uses the default TLS Context.

- When a TLS connection with the device is initiated by a SIP client, the device also responds using TLS, regardless of whether or not TLS was configured.
- The device regulates the number of new concurrent TLS connections that can be established per second. This protects the device from flooding (avalanches) of new TLS connections which may be caused from TLS-based malicious attacks or distributed denial-of-service (DDoS) attacks.
- To configure two-way (mutual) TLS authentication, see TLS for SIP Clients on the next page.

➢ **To configure SIPS:**

1. Configure a TLS Context (see Configuring TLS Certificate Contexts).
2. Assign the TLS Context to a Proxy Set or SIP Interface (see Configuring Proxy Sets and Configuring SIP Interfaces, respectively).
3. Configure the SIP Interface with a TLS port number.
4. Configure various SIPS parameters in the Security Settings page (**Setup** menu > **IP Network** tab > **Security** folder > **Security Settings**). For a description of the below TLS parameters, see TLS Parameters.



5. By default, the device initiates a TLS connection only for the next network hop. To enable TLS all the way to the destination (**over multiple hops**), open the Transport Settings page (**Setup** menu > **Signaling & Media** tab > **SIP Definitions** folder > **Transport Settings**), and then configure the 'SIPS' [EnableSIPS] parameter to **Enable**:

Configuring Mutual TLS Authentication

This section describes how to configure mutual (two-way) TLS authentication.

TLS for SIP Clients

When Secure SIP (SIPS) is implemented using TLS, it is sometimes required to use two-way (mutual) authentication between the device and a SIP user agent (client). When the device acts as the TLS server in a specific connection, the device demands the authentication of the SIP client's certificate. Both the device and the client use certificates from a CA to authenticate each other, sending their X.509 certificates to one another during the TLS handshake. Once the sender is verified, the receiver sends its' certificate to the sender for verification. SIP signaling starts when authentication of both sides completes successfully.

TLS mutual authentication can be configured for calls by enabling mutual authentication on the SIP Interface associated with the calls. The TLS Context associated with the SIP Interface or Proxy Set belonging to these calls are used.

> **Note:** SIP mutual authentication can also be configured globally for all calls, using the 'TLS Mutual Authentication' (SIPSRequireClientCertificate) parameter (see Configuring TLS for SIP).

➢ **To configure mutual TLS authentication for SIP messaging:**

1. Enable two-way authentication on the specific SIP Interface: In the SIP Interfaces table (see Configuring SIP Interfaces), configure the 'TLS Mutual Authentication' parameter to **Enable** for the specific SIP Interface.
2. Configure a TLS Context with the following certificates:
   - Import the certificate of the CA that signed the certificate of the SIP client into the Trusted Certificates table (certificate root store) so that the device can authenticate the client (see Importing Certificates into Trusted Root Certificate Store).
   - Make sure that the TLS certificate is signed by a CA that the SIP client trusts so that the client can authenticate the device.

TLS for Remote Device Management

For a description of secured device management by mutual TLS authentication, see Configuring Secured (HTTPS) Web on page 64.

Configuring Firewall Rules

The Firewall table lets you configure up to 500 firewall rules, which define network traffic filtering rules (*access list*) for incoming (ingress) traffic. The access list offers the following firewall possibilities:

- Blocking traffic from known malicious sources
- Allowing traffic only from known "friendly" sources, while blocking all other traffic
- Mixing allowed and blocked network sources
- Limiting traffic to a user-defined rate (blocking the excess)
- Limiting traffic to specific protocols and specific port ranges on the device

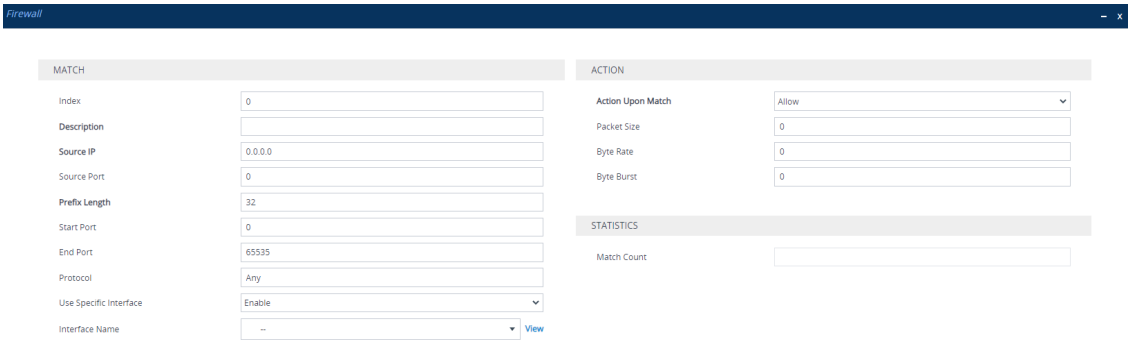
For each packet received on the network interface, the device searches the table from top to bottom until the first matching rule is found. The matched rule can permit (*allow*) or deny (*block*) the packet. Once a rule in the table is located, subsequent rules further down the table are ignored. If the end of the table is reached without a match, the packet is accepted.

- The rules configured by the Firewall table apply to a very low-level network layer and overrides all other security-related configuration. Thus, if you have configured higher-level security features (e.g., on the Application level), you must also configure firewall rules to permit this necessary traffic. For example, if you have configured IP addresses to access the device's Web and Telnet management interfaces in the Access List table (see Configuring Web and Telnet Access List), you must configure a firewall rule that permits traffic from these IP addresses.
- Only users with Security Administrator or Master access levels can configure firewall rules.
- The device supports dynamic firewall pinholes for media (RTP/RTCP) traffic negotiated in the SDP offer-answer of SIP calls. The pinhole allows the device to ignore its firewall and accept the traffic on the negotiated port. The device automatically closes the pinhole once the call terminates. Therefore, it is unnecessary to configure specific firewall rules to allow traffic through specific ports. For example, if you have configured a firewall rule to block all media traffic in the port range 6000 to 7000 and a call is negotiated to use the local port 6010, the device automatically opens port 6010 to allow the call.
- Setting the 'Prefix Length' field to **0** means that the rule applies to **all** packets, regardless of the defined IP address in the 'Source IP' field. Thus, it is highly recommended to set the parameter to a value other than 0.
- It is recommended to add a rule at the end of your table that blocks all traffic and to add firewall rules above it that allow required traffic (with bandwidth limitations). To block all traffic, use the following firewall rule:
  - ✓ Source IP: 0.0.0.0
  - ✓ Prefix Length: 0 (i.e., rule matches all IP addresses)
  - ✓ Start Port - End Port: 0-65535
  - ✓ Protocol: **Any**
  - ✓ Action Upon Match: **Block**
- If the device needs to communicate with AudioCodes OVOC, you must also add rules to allow incoming traffic from OVOC. For more information, see Configuring Firewall Rules to Allow Incoming OVOC Traffic on page 184.

The following procedure describes how to configure firewall rules through the Web interface. You can also configure it through ini file [AccessList] or CLI (`configure network > access-list`).

➢ **To configure a firewall rule:**

1. Open the Firewall table (**Setup** menu > **IP Network** tab > **Security** folder> **Firewall**).
2. Click **New**; the following dialog box appears:



3. Configure a firewall rule according to the parameters described in the table below.
4. Click **Apply**, and then reset the device with a save-to-flash for your settings to take effect.

Table 14-2: Firewall Table Parameter Descriptions

Parameter	Description
Match	
'Index'	Defines an index number for the new table row. **Note:** Each row must be configured with a unique index.
'Description' `description` [AccessList_Description]	Defines a brief description for the rule.
'Source IP' `source-ip` [AccessList_Source_IP]	Defines the IP address (or DNS name) or a specific host name of the source network from where the device receives the incoming packet. The default is 0.0.0.0.
'Source Port' `src-port` [AccessList_Source_Port]	Defines the source UDP/TCP ports of the remote host from where the device receives the incoming packet. The valid range is 0 to 65535. The default is 0. **Note:** ■ When set to 0, this field is ignored and any source port matches the rule. ■ The source ports used for outgoing TCP and TLS connections are not configurable and are dynamically determined by the device in the range of 32,768-61,000.
'Prefix Length' `prefixLen` [AccessList_PrefixLen]	(**Mandatory**) Defines the IP network mask of the IP address configured in the 'Source IP' parameter. ■ A value of 8 corresponds to IPv4 subnet class A

Parameter	Description
	(network mask of 255.0.0.0). ■ A value of 16 corresponds to IPv4 subnet class B (network mask of 255.255.0.0). ■ A value of 24 corresponds to IPv4 subnet class C (network mask of 255.255.255.0). ■ A value of 32 means a single host (network mask 255.255.255.255). In other words, the specific IP address in the 'Source IP' parameter. The IP address of the sender of the incoming packet is trimmed in accordance with the prefix length (in bits) and then compared to the IP address in the 'Source IP' parameter. The default is 32. **Note:** A value of 0 applies to **all** packets, regardless of the configured IP address ('Source IP'). Therefore, you must set the parameter to a value other than 0.
'Start Port' `start-port` [AccessList_Start_Port]	Defines the first UDP/TCP port in the range of ports on the device on which the incoming packet is received. From the perspective of the remote IP entity, this is the destination port. To configure the last port in the range, see the 'End Port' parameter (below). The valid range is 0 to 65535. **Note:** When the protocol type isn't TCP or UDP, the entire range must be provided.
'End Port' `end-port` [AccessList_End_Port]	Defines the last UDP/TCP port in the range of ports on the device on which the incoming packet is received. From the perspective of the remote IP entity, this is the destination port. To configure the first port in the range, see the 'Start Port' parameter (above). The valid range is 0 to 65535 (default). **Note:** When the protocol type isn't TCP or UDP, the entire range must be provided.
'Protocol' `protocol` [AccessList_Protocol]	Defines the protocol type (e.g., **UDP**, **TCP**, **ICMP**, **ESP** or **Any**) or the IANA protocol number in the range of 0 (Any) to 255. The default is **Any**. **Note:** ■ The parameter also accepts the string value "HTTP",

Parameter	Description
	which implies selection of the TCP or UDP protocols and the appropriate port numbers as defined on the device. ■ To specify SIP ports, configure rules with the UDP and TCP protocols for the required SIP Interfaces.
'Use Specific Interface' `use-specific-interface` [AccessList_Use_Specific_Interface]	Defines if you want to apply the rule to a specific network interface defined in the IP Interfaces table (i.e., packets received from that defined in the Source IP field and received on this network interface): ■ [0] **Disable** = The rule applies to all interfaces. ■ [1] **Enable** = (Default) The rule applies to a specific interface as specified in the 'Interface Name' field (see below).
'Interface Name' `network-interface-name` [AccessList_Interface_x]	Defines the network interface to which you want to apply the rule. This is applicable if you enabled the 'Use Specific Interface' field. The list displays interface names as defined in the IP Interfaces table in Configuring IP Network Interfaces.
Action	
'Action Upon Match' `allow-type` [AccessList_Allow_Type]	Defines the firewall action to be performed upon rule match. ■ **Allow** = (Default) Permits the packets. ■ **Block** = Rejects the packets
'Packet Size' `packet-size` [AccessList_Packet_Size]	Defines the maximum allowed packet size. The valid range is 0 to 65535. **Note:** When filtering fragmented IP packets, this field relates to the overall (re-assembled) packet size, and not to the size of each fragment.
'Byte Rate' `byte-rate` [AccessList_Byte_Rate]	Defines the expected traffic rate (bytes per second), i.e., the allowed bandwidth for the specified protocol. In addition to this field, the 'Burst Bytes' field provides additional allowance such that momentary bursts of data may utilize more than the defined byte rate, without being interrupted. For example, if 'Byte Rate' is set to 40000 and 'Burst

Parameter	Description
	Bytes' to 50000, then this implies the following: the allowed bandwidth is 40000 bytes/sec with extra allowance of 50000 bytes; if, for example, the actual traffic rate is 45000 bytes/sec, then this allowance would be consumed within 10 seconds, after which all traffic exceeding the allocated 40000 bytes/sec is dropped. If the actual traffic rate then slowed to 30000 bytes/sec, then the allowance would be replenished within 5 seconds.
'Burst Bytes' `byte-burst` [AccessList_Byte_Burst]	Defines the tolerance of traffic rate limit (number of bytes). The default is 0.
Statistics	
'Match Count' [AccessList_MatchCount]	(Read-only) Displays the number of packets accepted or rejected by the rule.

The table below provides an example of configured firewall rules:

Table 14-3: Configuration Example of Firewall Rules

Parameter	Firewall Rule				
	1	2	3	4	5
'Source IP'	12.194.231.76	12.194.230.7	0.0.0.0	192.0.0.0	0.0.0.0
'Prefix Length'	16	16	0	8	0
'Start Port and End Port'	0-65535	0-65535	0-65535	0-65535	0-65535
'Protocol'	Any	Any	icmp	Any	Any
'Use Specific Interface'	**Enable**	**Enable**	**Disable**	**Enable**	**Disable**
'Interface Name'	WAN	WAN	None	Voice-Lan	None
'Byte Rate'	0	0	40000	40000	0

Parameter	Firewall Rule				
'Burst Bytes'	0	0	50000	50000	0
'Action Upon Match'	**Allow**	**Allow**	**Allow**	**Allow**	**Block**

The firewall rules in the above configuration example do the following:

- **Rules 1 and 2:** Typical firewall rules that allow packets ONLY from specified IP addresses (e.g., proxy servers). Note that the prefix length is configured.
- **Rule 3:** A more "advanced" firewall rule - bandwidth rule for ICMP, which allows a maximum bandwidth of 40,000 bytes/sec with an additional allowance of 50,000 bytes. If, for example, the actual traffic rate is 45,000 bytes/sec, then this allowance would be consumed within 10 seconds, after which all traffic exceeding the allocated 40,000 bytes/sec is dropped. If the actual traffic rate then slowed to 30,000 bytes/sec, the allowance would be replenished within 5 seconds.
- **Rule 4:** Allows traffic from the LAN voice interface and limits bandwidth.
- **Rule 5:** Blocks all other traffic.

Configuring Firewall Rules to Allow Incoming OVOC Traffic

If the device needs to communicate with AudioCodes OVOC, you need to configure the device's firewall (Firewall table) with the below "allow" firewall rules to permit incoming traffic from OVOC.

These OVOC-related firewall rules are required only if have configured other various firewall rules. If you are not using the device's firewall, the device allows all traffic by default and the below firewall configuration is not required.

Table 14-4: Firewall Rules to Allow Traffic from OVOC

Index	Source IP	Source Port	Prefix Length	Start Port	End Port	Protocol	Use Specific Interface	Interface Name	Action Upon Match	Packet Size	Byte Rate	Byte Burst
0	Various rules for basic traffic.											
...												
N												
N+1 (SNMP)	<OVOC IP address>	1161	32	161	161	udp	Enable	OAM_IF	Allow	0	0	0
N+2 (NTP)	<OVOC IP address>	123	32	0	0	udp	Enable	<interface configured for NTP>	Allow	0	0	0
N+3 (HTTP)	<OVOC IP address>	80	32	0	0	tcp	Enable	<interface configured for file transfer>	Allow	0	0	0
N+4 (HTTPS)	<OVOC IP address>	443	32	0	0	tcp	Enable	<interface configured for file transfer>	Allow	0	0	0
N+5 (Qo	<OVOC IP	5000	32	0	0	tcp	Enable	<interface	Allow	0	0	0

Index	Source IP	Source Port	Prefix Length	Start Port	End Port	Protocol	Use Specific Interface	Interface Name	Action Upon Match	Packet Size	Byte Rate	Byte Burst
E)	address>							configured for QoE>				
N+6 (QoE-secured)	<OVOC IP address>	5001	32	0	0	tcp	Enable	<interface configured for QoE>	Allow	0	0	0
N+7 (default - drop)	0.0.0.0	0	0	0	65535	Any	Disable	--	Block	0	0	0

Firewall Rule to Allow Incoming Azure Load Balancer Traffic

When the device is deployed on the Microsoft Azure cloud platform and you have configured the device (Signaling Component) with firewall rules where the last rule is a Block rule for all traffic, you must add a rule to allow keep-alive traffic between the Azure Load Balancer health probe and the device. Therefore, in the Firewall table, add a rule before the Block rule with the following parameter settings:

- 'Source IP': 168.63.129.16
- 'Source Port': 0

- 'Prefix Length': 32
- 'Start Port': 315
- 'End Port': 315
- 'Protocol': TCP
- 'Use Specific Interface': **Disable**
- 'Action Upon Match': **Allow**
- 'Packet Size': 0
- 'Byte Rate': 0
- 'Byte Burst': 0

- This section is applicable only to Mediant CE.
- The 'Source IP' parameter configures the source IP address from where Load Balancer health probes originate. To make sure that the IP address (168.63.129.16) specified above is correct, go to Microsoft's web page on health probes.

Intrusion Detection System

The device's Intrusion Detection System (IDS) feature detects malicious attacks on the device and reacts accordingly. A remote host is considered malicious if it reaches or exceeds a user-defined threshold (counter) of specified malicious attack types.

If malicious activity is detected, the device can do the following:

- Block (blacklist) remote hosts (IP addresses / ports) considered by the device as malicious. The device automatically blacklists the malicious source for a user-defined period, after which it is removed from the blacklist.
- Send SNMP traps to notify of malicious activity and/or whether an attacker has been added to or removed from the blacklist. For more information, see Viewing IDS Alarms.

IDS is an important feature as it ensures legitimate calls are not being adversely affected by attacks, and prevents Theft of Service and unauthorized access.

There are many types of malicious attacks, the most common being:

- **Denial of service:** This can be Denial of Service (DoS) where an attacker wishing to prevent a server from functioning correctly directs a large amount of requests – sometimes meaningless and sometimes legitimate, or it can be Distributed Denial of Service (DDoS) where the attacker controls a large group of systems to coordinate a large scale DoS attack against a system:
  - Message payload tampering: Attacker may inject harmful content into a message, e.g., by entering meaningless or wrong information, with the goal of exploiting a buffer

overflow at the target. Such messages can be used to probe for vulnerabilities at the target.

- Message flow tampering: This is a special case of DoS attacks. These attacks disturb the ongoing communication between users. An attacker can then target the connection by injecting fake signaling messages into the communication channel (such as CANCEL messages).
- Message Flooding: The most common DoS attack is where an attacker sends a huge amount of messages (e.g., INVITEs) to a target. The goal is to overwhelm the target's processing capabilities, thereby rendering the target inoperable.

- **SPAM over Internet Telephony (SPIT):** VoIP spam is unwanted, automatically dialed, pre-recorded phone calls using VoIP. It is similar to e-mail spam.
- **Theft of Service (ToS):** Service theft can be exemplified by phreaking, which is a type of hacking that steals service (i.e., free calls) from a service provider, or uses a service while passing the cost to another person.

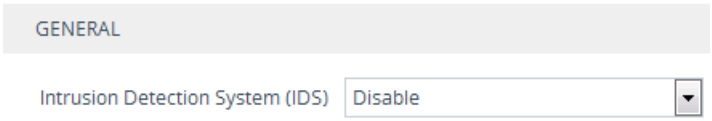
The IDS configuration is based on IDS Policies, where each policy can be configured with a set of IDS rules. Each rule defines a type of malicious attack to detect and the number of attacks during an interval (threshold) before an SNMP trap is sent. Each policy is then applied to a target under attack (SIP Interface) and/or source of attack (Proxy Set and/or subnet address).

Enabling IDS

By default, IDS is disabled. You can enable it, as described below.

➢ **To enable IDS:**

1. Open the IDS General Settings page (**Setup** menu > **Signaling & Media** tab > **Intrusion Detection** folder >**IDS General Settings**).



2. From the 'Intrusion Detection System' drop-down list, select **Enable**.
3. Click **Apply**.

Configuring IDS Policies

An IDS Policy is configured using two tables with "parent-child" type relationship:

- **IDS Policies table ("parent"):** Defines a name and provides a description for the IDS Policy. You can configure up to 20 IDS Policies.
- **IDS Rules table ("child"):** Defines the actual rules for the IDS Policy. Each IDS Policy can be configured with up to 20 rules.

> A maximum of 100 IDS rules can be configured (regardless of how many rules are assigned to each policy).

The device provides the following pre-configured IDS Policies that can be used in your deployment (if they meet your requirements):

- "DEFAULT_FEU": IDS Policy for far-end users in the WAN
- "DEFAULT_PROXY": IDS Policy for proxy server
- "DEFAULT_GLOBAL": IDS Policy with global thresholds

> - You can edit and delete the default IDS Policies.
> - If the IDS Policies table is empty (i.e., you have deleted all IDS Policies) and you want to return the default IDS Policies, disable IDS and then enable it again.

The following procedure describes how to configure IDS Policies through the Web interface. You can also configure it through ini file or CLI:

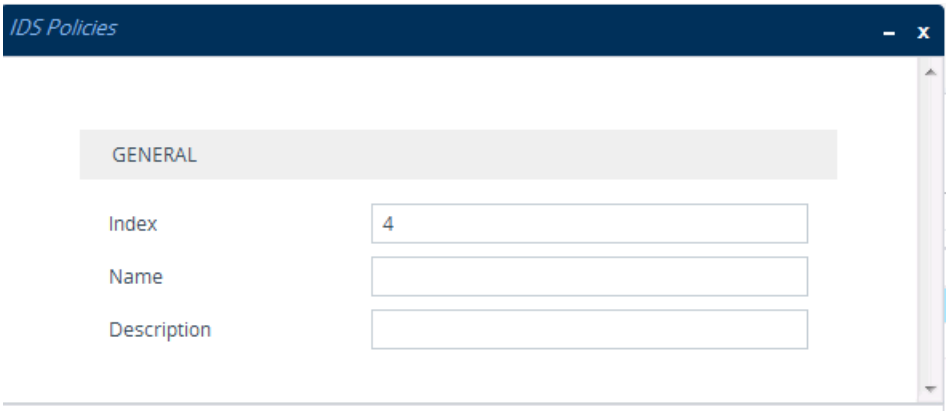
- IDS Policy table: IDSPolicy (ini file) or configure voip > ids policy (CLI)
- IDS Rules table: IDSRule (ini file) or configure voip > ids rule (CLI)

➢ To configure an IDS Policy:

1. Open the IDS Policies table (**Setup** menu > **Signaling & Media** tab > **Intrusion Detection** folder > **IDS Policies**); the table displays the pre-configured IDS policies:

INDEX	NAME	DESCRIPTION
0	DOS	dos-attacks
1	DEFAULT_FEU	Default policy for FEU
2	DEFAULT_PROXY	Default policy for proxies
3	DEFAULT_GLOBAL	Default policy for global scope

2. Click **New**; the following dialog box appears:

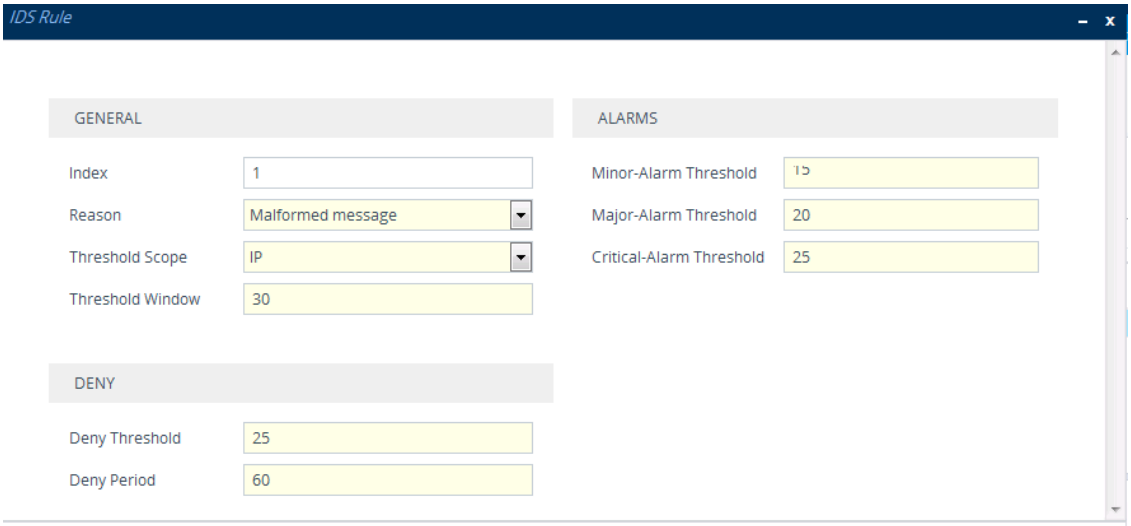


3. Configure an IDS Policy name according to the parameters described in the table below.
4. Click **Apply**.

Table 14-5: IDS Policies Table Parameter Descriptions

Parameter	Description
'Index' `policy` [IDSPolicy_Index]	Defines an index number for the new table row. **Note:** Each row must be configured with a unique index.
'Name' `rule` [IDSPolicy_Name]	Defines a descriptive name, which is used when associating the row in other tables. The valid value is a string of up to 40 characters. **Note:** The parameter value cannot contain a forward slash (/).
'Description' `description` [IDSPolicy_ Description]	Defines a brief description for the IDS Policy. The valid value is a string of up to 100 characters.

5. In the IDS Policies table, select the required IDS Policy row, and then click the **IDS Rule** link located below the table; the IDS Rule table opens.
6. Click **New**; the following dialog box appears:



The figure above shows a configuration example: If 15 malformed SIP messages ('Reason') are received within a period of 30 seconds ('Threshold Window'), a minor alarm is sent ('Minor-Alarm Threshold'). Every 30 seconds, the rule's counters are cleared ('Threshold Window'). If more than 25 malformed SIP messages are received within this period, the device blacklists for 60 seconds the remote IP host ('Deny Threshold') from where the messages were received.

7. Configure an IDS Rule according to the parameters described in the table below.
8. Click **Apply**, and then save your settings to flash memory.

Table 14-6: IDS Rule Table Parameter Descriptions

Parameter	Description
General	
'Index' `rule-id` [IDSRule_RuleID]	Defines an index number for the new table record.
'Reason' `reason` [IDSRule_Reason]	Defines the type of intrusion attack (malicious event). ■ [0] **Any** = All events listed below are considered as attacks and are counted together. ■ [1] **Connection abuse** = (Default) Connection failures, which includes the following: ✓ Incoming TLS authentication (handshake) failure ✓ Incoming WebSocket connection establishment failure ■ [2] **Malformed message** = Malformed SIP messages, which includes the following: ✓ Message exceeds a user-defined maximum message length (50K) ✓ Any SIP parser error ✓ Message Policy match (see Configuring SIP Message Policy Rules) ✓ Basic headers not present ✓ Content length header not present (for TCP) ✓ Header overflow ■ [3] **Authentication failure** = SIP authentication failure, which includes the following: ✓ Local authentication ("Bad digest" errors) ✓ Remote authentication (SIP 401/407 is sent if original message includes authentication) ■ [4] **Dialog establish failure** = SIP dialog establishment (e.g., INVITE) failure, which includes the following: ✓ Classification failure (see Configuring Classification Rules). ✓ Call Admission Control (CAC) threshold exceeded

Parameter	Description
	(see Configuring Call Admission Control on page 696) ✓ Routing failure (i.e., no routing rule was matched) ✓ Local reject by device (prior to SIP 180 response): REGISTER not allowed due to IP Group's 'RegistrationMode' parameter, or SIP requests rejected based on a registered users policy (configured by the SRD_BlockUnRegUsers or SIPInterface_BlockUnRegUsers parameters). ✓ No user found when routing to a User-type IP Group (similar to a SIP 404) ✓ Remote rejects (prior to SIP 18x response). To specify SIP response codes to exclude from the IDS count, see Configuring SIP Response Codes to Exclude from IDS on page 199. ✓ Malicious signature pattern detected (see Configuring Malicious Signatures) ■ [5] **Abnormal flow** = SIP call flow that is abnormal, which includes the following: ✓ Requests and responses without a matching transaction user (except ACK requests) ✓ Requests and responses without a matching transaction (except ACK requests)
'Threshold Scope' `threshold-scope` [IDSRule_ThresholdScope]	Defines the source of the attacker to consider in the device's detection count. ■ [0] **Global** = All attacks regardless of source are counted together during the threshold window. ■ [2] **IP** = Attacks from each specific IP address are counted separately during the threshold window. ■ [3] **IP+Port** = Attacks from each specific IP address:port are counted separately during the threshold window. This option is useful for NAT servers, where numerous remote machines use the same IP address but different ports. However, it is not recommended to use this option as it may degrade detection capabilities.
'Threshold Window' `threshold-window`	Defines the threshold interval (in seconds) during which the device counts the attacks to check if a threshold is crossed.

Parameter	Description
[IDSRule_ ThresholdWindow]	The counter is automatically reset at the end of the interval. The valid range is 1 to 1,000,000. The default is 1.
Alarms	
'Minor-Alarm Threshold' minor-alrm-thr [IDSRule_ MinorAlarmThreshold]	Defines the threshold that if crossed a minor severity alarm is sent. The valid range is 1 to 1,000,000. A value of 0 or -1 means not defined.
'Major-Alarm Threshold' major-alrm-thr [IDSRule_ MajorAlarmThreshold]	Defines the threshold that if crossed a major severity alarm is sent. The valid range is 1 to 1,000,000. A value of 0 or -1 means not defined.
'Critical-Alarm Threshold' critical-alrm-thr [IDSRule_ CriticalAlarmThreshold]	Defines the threshold that if crossed a critical severity alarm is sent. The valid range is 1 to 1,000,000. A value of 0 or -1 means not defined.
Deny	
'Deny Threshold' deny-thr [IDSRule_DenyThreshold]	Defines the threshold that if crossed, the device blocks (blacklists) the remote host (attacker). The default is -1 (i.e., not configured). **Note:** The parameter is applicable only if the 'Threshold Scope' parameter is set to **IP** or **IP+Port**.
'Deny Period' deny-period [IDSRule_DenyPeriod]	Defines the duration (in sec) to keep the attacker on the blacklist, if configured using the 'Deny Threshold' parameter. The valid range is 0 to 1,000,000. The default is -1 (i.e., not configured). **Note:** The parameter is applicable only if the 'Threshold Scope' parameter is set to **IP** or **IP+Port**.

Assigning IDS Policies

The IDS Matches table lets you implement your configured IDS Policies. You do this by assigning IDS Policies to any, or a combination of the following configuration entities:

- **SIP Interface:** For detection of malicious attacks on specific SIP Interface(s). To configure SIP Interfaces, see Configuring SIP Interfaces.

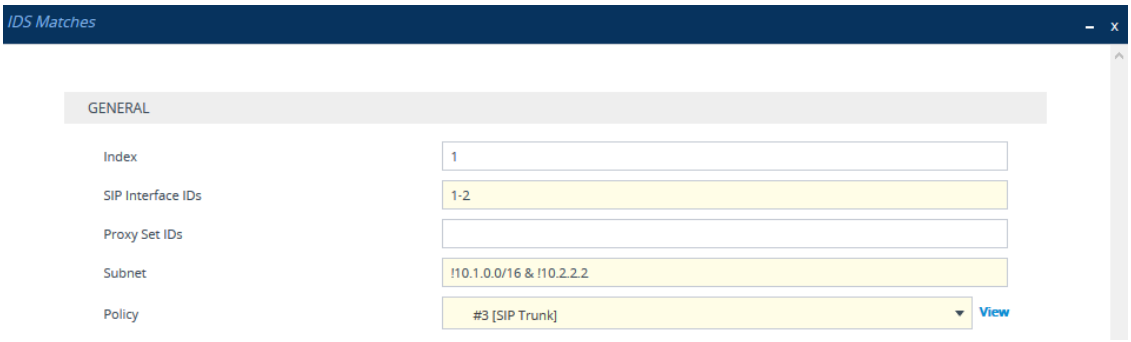
- **Proxy Sets:** For detection of malicious attacks from specified Proxy Set(s). To configure Proxy Sets, see Configuring Proxy Sets.
- **Subnet addresses:** For detection of malicious attacks from specified subnet addresses.

You can configure up to 20 IDS Policy-Matching rules.

The following procedure describes how to configure the IDS Match table through the Web interface. You can also configure it through ini file [IDSMatch] or CLI (`configure voip > ids match`).

➢ To configure an IDS Policy-Matching rule:

1. Open the IDS Matches table (**Setup** menu > **Signaling & Media** tab > **Intrusion Detection** folder > **IDS Matches**).
2. Click **New**; the following dialog box appears:



The figure above shows a configuration example where the IDS Policy "SIP Trunk" is applied to SIP Interfaces 1 and 2, and to all source IP addresses outside of subnet 10.1.0.0/16 and IP address 10.2.2.2.

3. Configure a rule according to the parameters described in the table below.
4. Click **Apply**, and then save your settings to flash memory.

Table 14-7: IDS Matches Table Parameter Descriptions

Parameter	Description
'Index' [IDSMatch_Index]	Defines an index number for the new table record.
'SIP Interface IDs' `sip-interface` [IDSMatch_SIPInterface]	Assigns a SIP Interface(s) to the IDS Policy. This indicates the SIP Interfaces that are being attacked. The valid value is the ID of the SIP Interface. The following syntax is supported: ■ A comma-separated list of SIP Interface IDs (e.g., 1,3,4) ■ A hyphen (-) indicates a range of SIP Interfaces (e.g., 3,4-7 means IDs 3, and 4 through 7)

Parameter	Description
	■ A prefix of an exclamation mark (!) means negation of the set (e.g., !3,4-7 means all indexes excluding 3, and excluding 4 through 7)
'Proxy Set IDs' `proxy-set` [IDSMatch_ProxySet]	Assigns a Proxy Set(s) to the IDS Policy. This indicates the Proxy Sets from where the attacks are coming from. The following syntax is supported: ■ A comma-separated list of Proxy Set IDs (e.g., 1,3,4) ■ A hyphen (-) indicates a range of Proxy Sets (e.g., 3,4-7 means IDs 3, and 4 through 7) ■ A prefix of an exclamation mark (!) means negation of the set (e.g., !3,4-7 means all indexes excluding 3, and excluding 4 through 7) **Note:** ■ Only the IP address of the Proxy Set is considered (not port). ■ If a Proxy Set has multiple IP addresses, the device considers the Proxy Set as one entity and includes all its IP addresses in the same IDS count.
'Subnet' `subnet` [IDSMatch_Subnet]	Defines the subnet to which the IDS Policy is assigned. This indicates the subnets from where the attacks are coming from. The following syntax can be used: ■ Basic syntax is a subnet in CIDR notation (e.g., 10.1.0.0/16 means all sources with IP address in the range 10.1.0.0–10.1.255.255) ■ An IP address can be specified without the prefix length to refer to the specific IP address. ■ Each subnet can be negated by prefixing it with (!), which means all IP addresses outside that subnet. ■ Multiple subnets can be specified by separating them with "&" (and) or "|" (or) operations (without quotation marks), for example: ✓ 10.1.0.0/16 | 10.2.2.2: includes subnet 10.1.0.0/16 and IP address 10.2.2.2. ✓ !10.1.0.0/16 & !10.2.2.2: includes all addresses except those of subnet 10.1.0.0/16 and IP address 10.2.2.2. Note that the exclamation mark (!) appears before

Parameter	Description
	each subnet. ✓ 10.1.0.0/16 & !10.1.1.1: includes subnet 10.1.0.0/16, except IP address 10.1.1.1.
'Policy' policy [IDSMatch_Policy]	Assigns an IDS Policy (configured in Configuring IDS Policies).

Viewing IDS Alarms

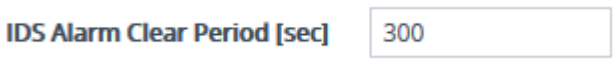
For the IDS feature, the device sends the following SNMP traps:

- Traps that notify the detection of malicious attacks:
  - **acIDSPolicyAlarm:** The device sends this alarm whenever a threshold of a specific IDS Policy rule is crossed. The trap displays the crossed severity threshold (Minor or Major), IDS Policy and IDS Rule, and the IDS Policy-Match index.
  - **acIDSThresholdCrossNotification:** The device sends this event for each scope (IP address) that crosses the threshold. In addition to the crossed severity threshold (Minor or Major) of the IDS Policy-Match index, this event shows the IP address (or IP address:port) of the malicious attacker.

    If the severity level is raised, the alarm of the former severity is cleared and the device sends a new alarm with the new severity. The alarm is cleared after a user-defined timeout during which no thresholds have been crossed.

➢ To configure IDS alarm cleared timeout:

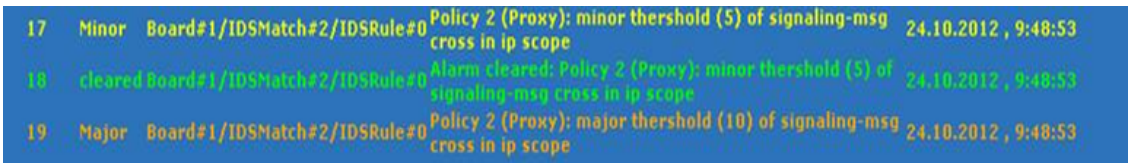
1. Open the IDS General Settings page (**Setup** menu > **Signaling & Media** tab > **Intrusion Detection** folder > **IDS General Settings**).
2. From the 'IDS Alarm Clear Period' field (IDSAlarmClearPeriod), enter the timeout (in seconds) after which the alarm is cleared if no IDS thresholds have been crossed during the timeout.



3. Click **Apply**.

However, this "quiet" timeout period must be at least twice the 'Threshold Window' value (configured in Configuring IDS Policies). For example, if you set IDSAlarmClearPeriod to 20 sec and 'Threshold Window' to 15 sec, the IDSAlarmClearPeriod parameter is ignored and the alarm is cleared only after 30 seconds (2 x 15 sec).

The figure below displays an example of IDS alarms in the Active Alarms table (Viewing Active Alarms). In this example, a Minor threshold alarm is cleared and replaced by a Major threshold alarm:



- acIDSBlacklistNotification event: The device sends this event whenever an attacker (remote host at IP address and/or port) is added to or removed from the blacklist.

You can also view IDS alarms through CLI:

- To view all active IDS alarms:

```
# show voip ids active-alarm all
```

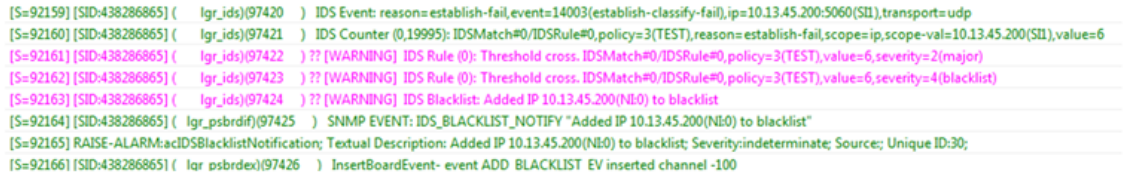
- To view all IP addresses that have crossed the threshold for an active IDS alarm:

```
# show voip ids active-alarm match <IDS Match Policy ID> rule <IDS Rule ID>
```

The IP address is displayed only if the 'Threshold Scope' parameter is set to IP or IP+Port; otherwise, only the alarm is displayed.

- To view the blacklist, see Viewing IDS Active Blacklist on page 975

The device also sends IDS notifications and alarms in Syslog messages to a Syslog server. This occurs only if you have configured Syslog (see Enabling Syslog). An example of a Syslog message with IDS alarms and notifications is shown below:



The table below lists the Syslog text messages per malicious event:

Table 14-8: Types of Malicious Events and Syslog Text String

Reason	
Description	**Syslog String**
Connection Abuse	
TLS authentication failure	abuse-tls-auth-fail
WebSocket establishment failure	abuse-websocket-fail

Reason	
Description	**Syslog String**
Malformed Messages	
Message exceeds a user-defined maximum message length (50K)	malformed-invalid-msg-len
Any SIP parser error	malformed-parse-error
Message policy match	malformed-message-policy
Basic headers not present	malformed-miss-header
Content length header not present (for TCP)	malformed-miss-content-len
Header overflow	malformed-header-overflow
Authentication Failure	
Local authentication ("Bad digest" errors)	auth-establish-fail
Remote authentication (SIP 401/407 is sent if original message includes authentication)	auth-reject-response
Dialog Establishment Failure	
Classification failure	establish-classify-fail
Routing failure (no matched routing rule)	establish-route-fail
Other local rejects (prior to SIP 180 response)	establish-local-reject
Remote rejects (prior to SIP 180 response)	establish-remote-reject
Malicious signature pattern detected	establish-malicious-signature-db-reject
CAC threshold exceeded	establish-cac-reject
Abnormal Flow	
Requests and responses without a matching transaction user (except ACK requests)	flow-no-match-tu
Requests and responses without a matching transaction (except ACK requests)	flow-no-match-transaction

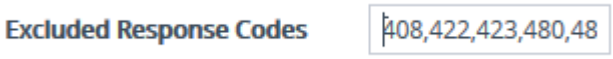
Configuring SIP Response Codes to Exclude from IDS

You can specify SIP response codes (reject reasons) that you want the IDS mechanism to ignore in its' count as reasons for SIP-dialog establishment failures.

➢ **To configure SIP responses to exclude from IDS:**

1. Open the IDS General Settings page (**Setup** menu > **Signaling & Media** tab > **Intrusion Detection** folder > **IDS General Settings**).
2. In the 'Excluded Response Codes' field, enter the SIP response codes that you want ignored by IDS.

Figure 14-1: Configuring SIP Response Codes to Exclude from IDS



3. Click **Apply**.

- The parameter applies only to rejected responses received from the upstream server; not rejected responses generated by the device (except for 404).
- The response codes 401 and 407 are considered authentication failures and thus, are not applicable to this parameter.

15 Media

This section describes media-related configuration.

Configuring Voice Settings

The section describes various voice-related configuration such as voice volume, silence suppression, and DTMF transport type. For a detailed description of these parameters, see Configuration Parameters Reference.

Configuring Voice Gain (Volume) Control

The device allows you to configure the level of the received (input gain) IP-to-IP signal and the level of the transmitted (output gain) IP-to-IP signal. The gain can be set between -32 and 31 decibels (dB).

➢ **To configure gain control through the Web interface:**

1. Open the Voice Settings page (**Setup** menu > **Signaling & Media** tab > **Media** folder > **Voice Settings**).

2. Configure the following parameters:
   - 'Voice Volume' (*VoiceVolume*)**:** Defines the voice gain control (in decibels) of the transmitted signal.
   - 'Input Gain' (*InputGain*): Defines the PCM input gain control (in decibels) of the received signal.
3. Click **Apply**.

Configuring Echo Cancellation

The device supports adaptive linear (line) echo cancellation according to G.168-2002. Echo cancellation is a mechanism that removes echo from the voice channel. Echoes are reflections of the transmitted signal.

In this line echo, echoes are generated when two-wire telephone circuits (carrying both transmitted and received signals on the same wire pair) are converted to a four-wire circuit. Echoes are reflections of the transmitted signal, which result from impedance mismatch in the hybrid (bi-directional 2-wire to 4-wire converting device).

An estimated echo signal is built by feeding the decoder output signal to an RLS-like adaptive filter, which adapts itself to the characteristics of the echo path. The 'estimated echo signal' (the output of this filter) is then subtracted from the input signal (which is the sum of the

desired input signal and the undesired echo) to provide a clean signal. To suppress the remaining residual echo, a Non Linear Processor (NLP) is used, as well as a double-talk (two people speak at the same time) detector that prevents false adaptation during near-end speech.

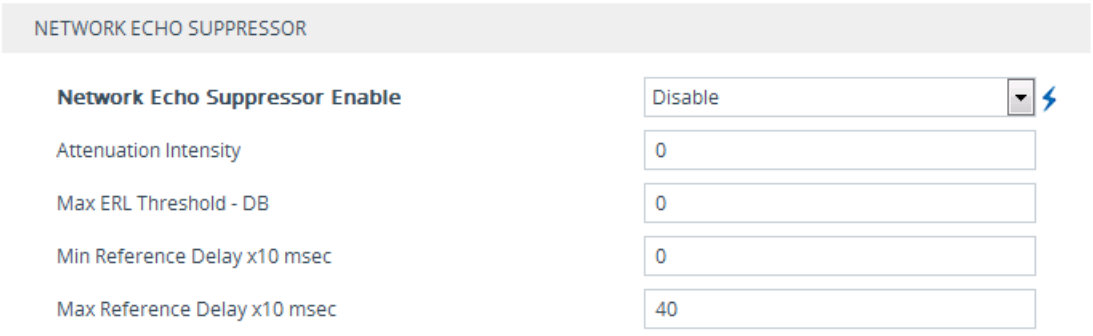
The device also supports acoustic echo cancellation for SBC calls. These echoes are composed of undesirable acoustical reflections (non-linear) of the received signal (i.e., from the speaker) which find their way from multiple reflections such as walls and windows into the transmitted signal (i.e., microphone). Therefore, the party at the far end hears his / her echo. The device removes these echoes and sends only the near-end's desired speech signal to the network (i.e., to the far-end party). The echo is composed of a linear part and a nonlinear part. However, in the Acoustic Echo Canceler, a substantial part of the echo is non-linear echo. To support this feature, the Forced Transcoding feature must be enabled so that the device uses DSPs.

The following procedure describes how to configure echo cancellation through the Web interface:

➢ To configure echo cancellation:

1. Configure line echo cancellation:
   a. Open the Voice Settings page (**Setup** menu > **Signaling & Media** tab > **Media** folder > **Voice Settings**).

   b. From the 'Echo Canceller' drop-down list (*EnableEchoCanceller*), select **Enable**.
2. Enable acoustic echo cancellation for SBC calls:
   a. Open the Voice Settings page (Setup menu > Signaling & Media tab > Media folder > Voice Settings).
   b. Under the Network Echo Suppressor group:



c. In the Voice Settings page, configure the following parameters:

- 'Network Echo Suppressor Enable' (AcousticEchoSuppressorSupport) - enables the network Acoustic Echo Suppressor
- 'Echo Canceller Type' (EchoCancellerType) - defines the echo canceller type
- 'Attenuation Intensity' (AcousticEchoSuppAttenuationIntensity) - defines the acoustic echo suppressor signals identified as echo attenuation intensity
- 'Max ERL Threshold' (AcousticEchoSuppMaxERLThreshold) - defines the acoustic echo suppressor maximum ratio between signal level and returned echo from the phone
- 'Min Reference Delay' (AcousticEchoSuppMinRefDelayx10ms) - defines the acoustic echo suppressor minimum reference delay
- 'Max Reference Delay' (AcousticEchoSuppMaxRefDelayx10ms) - defines the acoustic echo suppressor maximum reference delay

d. Open the IP Profiles table, and configure the 'Echo Canceller' parameter to Acoustic (see Configuring IP Profiles).

e. Enable the Forced Transcoding feature (using the TranscodingMode parameter) to allow the device to use DSP channels, which are required for acoustic echo cancellation.

The following additional echo cancellation parameters are configurable only through the *ini* file:
- *ECHybridLoss* **-** defines the four-wire to two-wire worst-case Hybrid loss
- *ECNLPMode* **-** defines the echo cancellation Non-Linear Processing (NLP) mode
- *EchoCancellerAggressiveNLP* **-** enables Aggressive NLP at the first 0.5 second of the call

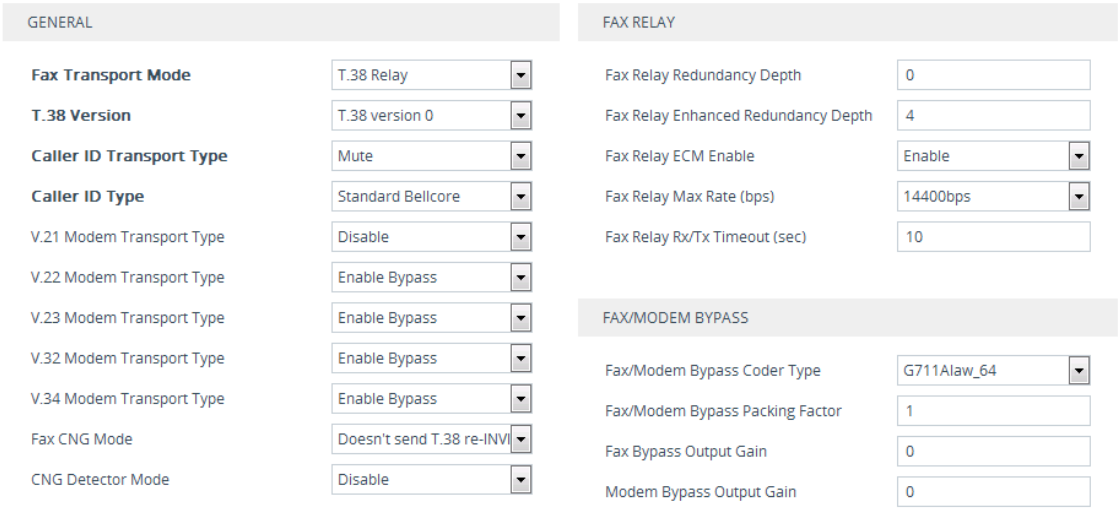
Fax and Modem Capabilities

This section describes the device's fax and modem capabilities and corresponding configuration. Fax and modem configuration is done on the Fax/Modem/CID Settings page.

- Unless otherwise specified, parameters mentioned in this section are available on this page. For a detailed description of these fax and modem parameters, see Configuration Parameters Reference.
- Some SIP parameters override these fax and modem parameters. For example, the [IsFaxUsed] parameter and V.152 parameters in Section V.152 Support.

➢ **To access the fax and modem parameters:**

■ Open the Fax/Modem/CID Settings page (**Setup** menu > **Signaling & Media** tab > **Media** folder > **Fax/Modem/CID Settings**).



Fax and Modem Operating Modes

The device supports two modes of operation:

- Fax/modem negotiation that is **not** performed during call establishment.
- Voice-band data (VBD) mode for V.152 implementation (see V.152 Support): Fax/modem capabilities are negotiated between the device and the remote endpoint during call establishment. During a call, when a fax/modem signal is detected, transition from voice to VBD (or T.38) is automatically performed and no additional SIP signaling is required. If negotiation fails (i.e., no match is achieved for any of the transport capabilities), fallback to existing logic occurs (according to the parameter IsFaxUsed).

Fax and Modem Transport Modes

The device supports the following transport modes for fax per modem type (V.22/V.23/Bell/V.32/V.34):

- T.38 fax relay (see T.38 Fax Relay Mode)
- G.711 Transport: switching to G.711 when fax/modem is detected (see G.711 Fax / Modem Transport Mode)
- Fax and modem bypass: a proprietary method that uses a high bit rate coder (see Fax/Modem Bypass Mode)
- NSE Cisco's Pass-through bypass mode for fax and modem (see Fax / Modem NSE Mode)
- Transparent with events: passing the fax / modem signal in the current voice coder with adaptations (see Fax / Modem Transparent with Events Mode)
- Transparent: passing the fax / modem signal in the current voice coder (see Fax / Modem Transparent Mode)
- RFC 2833 ANS Report upon Fax/Modem Detection (see RFC 2833 ANS Report upon Fax/Modem Detection)

'Adaptations' refer to automatic reconfiguration of certain DSP features for handling fax/modem streams differently than voice.

T.38 Fax Relay Mode

In Fax Relay mode, fax signals are transferred using the T.38 protocol. T.38 is the ITU standard for sending fax across IP networks in real-time mode. The device currently supports only the T.38 UDP syntax.

T.38 can be configured in the following ways:

- Switching to T.38 mode using SIP re-INVITE messages (see Switching to T.38 Mode using SIP Re-INVITE)
- Automatically switching to T.38 mode without using SIP re-INVITE messages (see Automatically Switching to T.38 Mode without SIP Re-INVITE)

When fax transmission ends, the reverse switching from fax relay to voice is automatically performed at both the local and remote endpoints.

You can change the fax rate declared in the SDP, using the 'Fax Relay Max Rate' parameter [FaxRelayMaxRate]. The parameter does not affect the actual transmission rate. You can also enable or disable Error Correction Mode (ECM) fax mode using the 'Fax Relay ECM Enable' parameter [FaxRelayECMEnable].

When using T.38 mode, you can define a redundancy feature to improve fax transmission over congested IP networks. This feature is activated using the 'Fax Relay Redundancy Depth' parameter [FaxRelayRedundancyDepth] and the 'Fax Relay Enhanced Redundancy Depth' parameter [FaxRelayEnhancedRedundancyDepth]. Although this is a proprietary redundancy scheme, it should not create problems when working with other T.38 decoders.

Switching to T.38 Mode using SIP re-INVITE

In the Switching to T.38 mode using the SIP re-INVITE, upon detection of a fax signal, the terminating device negotiates T.38 capabilities using a re-INVITE message. If the far-end device doesn't support T.38, the fax fails. In this mode, the 'Fax Transport Mode' parameter [FaxTransportMode] is ignored.

➢ **To configure T.38 mode using SIP re-INVITE messages:**

1. On the Fax/Modem/CID Settings page, configure the following optional parameters:
   - 'Fax Relay Redundancy Depth' [FaxRelayRedundancyDepth]
   - 'Fax Relay Enhanced Redundancy Depth' [FaxRelayEnhancedRedundancyDepth]
   - 'Fax Relay ECM Enable' [FaxRelayECMEnable]
   - 'Fax Relay Max Rate' [FaxRelayMaxRate]

The terminating gateway sends T.38 packets immediately after the T.38 capabilities are negotiated in SIP. However, the originating device by default, sends T.38 (assuming the T.38 capabilities are negotiated in SIP) only after it receives T.38 packets from the remote device. This default behavior cannot be used when the originating device is located behind a firewall that blocks incoming T.38 packets on ports that have not yet received T.38 packets from the internal network. To resolve this problem, you should configure the device to send CNG packets in T.38 upon CNG signal detection [CNGDetectorMode = 1].

Automatically Switching to T.38 Mode without SIP re-INVITE

In the Automatically Switching to T.38 mode without SIP re-INVITE, when a fax signal is detected, the channel automatically switches from the current voice coder to answer tone mode and then to T.38-compliant fax relay mode.

➢ **To configure Automatic T.38 mode:**

1. On the Fax/Modem/CID Settings page, configure the 'Fax Transport Mode' parameter to **T.38 Relay** [FaxTransportMode = 1].
2. Configure the following optional parameters:
   - 'Fax Relay Redundancy Depth' [FaxRelayRedundancyDepth]
   - 'Fax Relay Enhanced Redundancy Depth' [FaxRelayEnhancedRedundancyDepth]
   - 'Fax Relay ECM Enable' [FaxRelayECMEnable]
   - 'Fax Relay Max Rate' [FaxRelayMaxRate]

Fax over IP using T.38 Transmission over RTP

The device supports Fax-over-IP (FoIP) transmission using T.38 over RTP, whereby the T.38 payload is encapsulated in the RTP packet instead of being sent in dedicated T.38 packets (out-of-band). To support this feature, configure the coder type to T.38 Over RTP.

To indicate T.38 over RTP, the SDP body uses "udptl" (Facsimile UDP Transport Layer) in the 'a=ftmp' line. The device supports T.38 over RTP according to this standard and according to the AudioCodes proprietary method:

- **Call Parties belong to AudioCodes Devices:** T.38-over-RTP method is used, whereby the device encapsulates the entire T.38 packet (payload with all its headers) in the sent RTP. For T.38 over RTP, the devices use the proprietary identifier "AcUdptl" in the 'a=ftmp' line of the SDP. For example:

```
v=0
o=AudioCodesGW 1357424688 1357424660 IN IP4 10.8.6.68
s=Phone-Call
c=IN IP4 10.8.6.68
t=0 0
```

```
m=audio 6080 RTP/AVP 18 100 96
a=ptime:20
a=sendrecv
a=rtpmap:18 G729/8000
a=fmtp:18 annexb=no
a=rtpmap:100 t38/8000
a=fmtp:100 T38FaxVersion=0
a=fmtp:100 T38MaxBitRate=0
a=fmtp:100 T38FaxMaxBuffer=3000
a=fmtp:100 T38FaxMaxDatagram=122
a=fmtp:100 T38FaxRateManagement=transferredTCF
a=fmtp:100 T38FaxUdpEC=t38UDPRedundancy
a=fmtp:100 AcUdptl
a=rtpmap:96 telephone-event/8000
a=fmtp:96 0-15
```

- **AudioCodes Call Party with non-AudioCodes Party:** The device uses the standard T.38-over-RTP method, which encapsulates the T.38 payload only, without its headers (i.e., includes only fax data) in the sent RTP packet (RFC 4612).

The T.38-over-RTP method also depends on the initiator of the call:

- **Device initiates a call:** The device always sends the SDP offer with the proprietary token "AcUdpTl" in the 'fmtp' attribute. If the SDP answer includes the same token, the device employs the proprietary T.38-over-RTP mode; otherwise, the standard mode is used.
- **Device answers a call:** If the SDP offer from the remote party contains the 'fmtp' attribute with "AcUdpTl", the device answers with the same attribute and employs the proprietary T.38-over-RTP mode; otherwise, the standard mode is used.

If both T.38 (regular) and T.38 Over RTP coders are negotiated between the call parties, the device uses T.38 Over RTP.

G.711 Fax and Modem Transport Mode

In this mode, when the terminating device detects fax or modem signals (CED or AnsAM), it sends a re-INVITE message to the originating device, requesting it to re-open the channel in G.711 VBD with the following adaptations:

- Echo Canceller = off
- Silence Compression = off
- Echo Canceller Non-Linear Processor Mode = off
- Dynamic Jitter Buffer Minimum Delay = 40
- Dynamic Jitter Buffer Optimization Factor = 13

After a few seconds upon detection of fax V.21 preamble or super G3 fax signals, the device sends a second re-INVITE enabling the echo canceller (the echo canceller is disabled only on modem transmission).

A 'gpmd' attribute is added to the SDP according to the following format:

- **For G.711 A-law:**

```
a=gpmd:8 vbd=yes;ecan=on (or off for modems)
```

- **For G.711 µ-law:**

```
a=gpmd:0 vbd=yes;ecan=on (or off for modems)
```

The following parameters are ignored and automatically set to **Events Only**:

- 'Fax Transport Mode' [FaxTransportMode]
- 'Vxx ModemTransportType' [VxxModemTransportType]

Fax and Modem Bypass Mode

In this proprietary mode, when fax or modem signals are detected, the channel automatically switches from the current voice coder to a high bit-rate coder, according to the 'Fax/Modem Bypass Coder Type' parameter [FaxModemBypassCoderType]. The channel is also automatically reconfigured with the following fax / modem adaptations:

- Disables silence suppression
- Enables echo cancellation for fax
- Disables echo cancellation for modem
- Performs certain jitter buffering optimizations

The network packets generated and received during the bypass period are regular voice RTP packets (per the selected bypass coder), but with a different RTP payload type according to the following parameters:

- 'Fax Bypass Payload Type' [FaxBypassPayloadType]
- [ModemBypassPayloadType]

During the bypass period, the coder uses the packing factor, configured by the 'Fax/Modem Bypass Packing Factor' parameter [FaxModemBypassM]. The packing factor determines the number of coder payloads (each the size of [FaxModemBypassBasicRTPPacketInterval]) that are used to generate a single fax/modem bypass packet. When fax/modem transmission ends, the reverse switching, from bypass coder to regular voice coder is performed.

➢ **To configure fax / modem bypass mode:**

1. On the Fax/Modem/CID Settings page, do the following:

- Configure the 'Fax Transport Mode' parameter to **Bypass** [FaxTransportMode = 2].
- Configure the 'V.21 Modem Transport Type' parameter to **Enable Bypass** [V21ModemTransportType = 2].
- Configure the 'V.22 Modem Transport Type' parameter to **Enable Bypass** [V22ModemTransportType = 2].
- Configure the 'V.23 Modem Transport Type' parameter to **Enable Bypass** [V23ModemTransportType = 2].
- Configure the 'V.32 Modem Transport Type' parameter to **Enable Bypass** [V32ModemTransportType = 2].
- Configure the 'V.34 Modem Transport Type' parameter to **Enable Bypass** [V34ModemTransportType = 2].

2. Configure the [BellModemTransportType] parameter to 2 (Bypass).
3. Configure the following optional parameters:
   - 'Fax/Modem Bypass Coder Type' [FaxModemBypassCoderType]
   - 'Fax Bypass Payload Type' [FaxBypassPayloadType]
   - [ModemBypassPayloadType]
   - [FaxModemBypassBasicRTPPacketInterval]
   - [FaxModemBypasDJBufMinDelay]

- When the device is configured for modem bypass and T.38 fax, V.21 low-speed modems are not supported and fail as a result.
- When the remote (non-AudioCodes) gateway uses the G.711 coder for voice and doesn't change the coder payload type for fax or modem transmission, it is recommended to use the Bypass mode with the following configuration:
  - ✓ [EnableFaxModemInbandNetworkDetection = 1].
  - ✓ 'Fax/Modem Bypass Coder Type' = same coder used for voice.
  - ✓ 'Fax/Modem Bypass Packing Factor'[FaxModemBypassM] = same interval as voice.
  - ✓ [ModemBypassPayloadType = 8] if voice coder is A-Law or 0 if voice coder is Mu-Law.

Fax and Modem NSE Mode

In this mode, fax and modem signals are transferred using Cisco-compatible Pass-through bypass mode. Upon detection of fax or modem answering tone signal, the terminating device sends three to six special NSE RTP packets (configured by the [NSEpayloadType] parameter; usually to 100). These packets signal the remote device to switch to G.711 coder, according to the 'Fax/Modem Bypass Packing Factor' parameter. After a few NSE packets are exchanged between the devices, both devices start using G.711 packets with standard payload type (8 for

G.711 A-Law and 0 for G.711 Mu-Law). In this mode, no re-INVITE messages are sent. The voice channel is optimized for fax/modem transmission (same as for usual bypass mode).

The following parameters that configure the payload type for the AudioCodes proprietary Bypass mode are not used with NSE Bypass: 'Fax Bypass Payload Type' and [ModemBypassPayloadType].

When configured for NSE mode, the device includes the following line in the SDP, where *100* is the NSE payload type:

```
a=rtpmap:100 X-NSE/8000
```

The Cisco gateway must include the following definition:

```
modem passthrough nse payload-type 100 codec g711alaw
```

➢ **To configure NSE mode:**

1. On the Fax/Modem/CID Settings page, do the following:
   - Configure the 'Fax Transport Mode' parameter to **Bypass** [FaxTransportMode = 2].
   - Configure the 'V.21 Modem Transport Type' parameter to **Enable Bypass** [V21ModemTransportType = 2].
   - Configure the 'V.22 Modem Transport Type' parameter to **Enable Bypass** [V22ModemTransportType = 2].
   - Configure the 'V.23 Modem Transport Type' parameter to **Enable Bypass** [V23ModemTransportType = 2].
   - Configure the 'V.32 Modem Transport Type' parameter to **Enable Bypass** [V32ModemTransportType = 2].
   - Configure the 'V.34 Modem Transport Type' parameter to **Enable Bypass** [V34ModemTransportType = 2].
2. Configure the [BellModemTransportType] parameter to [2] (Bypass).
3. Configure the [NSEMode] parameter to [1] (enables NSE).
4. parameter the [NSEPayloadType] parameter to [100].

Fax and Modem Transparent with Events Mode

In this mode, fax and modem signals are transferred using the current voice coder with the following automatic adaptations:

- Echo Canceller = on (or off for modems)
- Echo Canceller Non-Linear Processor Mode = off
- Jitter buffering optimizations

➢ **To configure fax / modem transparent with events mode:**

1. On the Fax/Modem/CID Settings page, do the following:
   - Configure the 'Fax Transport Mode' parameter to **Events Only** [FaxTransportMode = 3].
   - Configure the 'V.21 Modem Transport Type' parameter to **Events Only** [V21ModemTransportType = 3].
   - Configure the 'V.22 Modem Transport Type' parameter to **Events Only** [V22ModemTransportType = 3].
   - Configure the 'V.23 Modem Transport Type' parameter to **Events Only** [V23ModemTransportType = 3].
   - Configure the 'V.32 Modem Transport Type' parameter to **Events Only** [V32ModemTransportType = 3].
   - Configure the 'V.34 Modem Transport Type' parameter to **Events Only** [V34ModemTransportType = 3].
2. Configure the [BellModemTransportType] parameter to [3] (transparent with events).

Fax / Modem Transparent Mode

In this mode, fax and modem signals are transferred using the current voice coder without notifications to the user and without automatic adaptations. It's possible to use Profiles (see Coders and Profiles) to apply certain adaptations to the channel used for fax / modem. For example, to use the coder G.711, to set the jitter buffer optimization factor to 13, and to enable echo cancellation for fax and disable it for modem.

➢ **To configure fax / modem transparent mode:**

1. On the Fax/Modem/CID Settings page, do the following:
   - Configure the 'Fax Transport Mode' parameter to **Disable** (FaxTransportMode = 0].
   - Configure the 'V.21 Modem Transport Type' parameter to **Disable** [V21ModemTransportType = 0].
   - Configure the 'V.22 Modem Transport Type' parameter to **Disable** [V22ModemTransportType = 0].
   - Configure the 'V.23 Modem Transport Type' parameter to **Disable** [V23ModemTransportType = 0].
   - Configure the 'V.32 Modem Transport Type' parameter to **Disable** [V32ModemTransportType = 0].
   - Configure the 'V.34 Modem Transport Type' parameter to **Disable** [V34ModemTransportType = 0].
2. Configure the [BellModemTransportType] parameter to [0] (transparent mode).
3. Configure the following optional parameters:

- Coders in the Coders table - see Configuring Coder Groups.
- 'Dynamic Jitter Buffer Optimization Factor' [DJBufOptFactor] - Configuring the Dynamic Jitter Buffer.
- 'Echo Canceller' [EnableEchoCanceller] - see Configuring Echo Cancellation.

This mode can be used for fax, but is not recommended for modem transmission. Instead, use the Bypass (see Fax/Modem Bypass Mode) or Transparent with Events modes (see Fax / Modem Transparent with Events Mode) for modem.

RFC 2833 ANS Report upon Fax and Modem Detection

The device (terminator gateway) sends RFC 2833 ANS/ANSam events upon detection of fax and/or modem answer tones (i.e., CED tone). This causes the originator to switch to fax/modem. The parameter is applicable only when the fax or modem transport type is set to bypass, Transparent-with-Events, V.152 VBD, or G.711 transport. When the device is located on the originator side, it ignores these RFC 2833 events.

➢ **To configure RFC 2833 ANS Report upon fax/modem detection:**

1. On the Fax/Modem/CID Settings page, do the following:
   - Configure the 'Fax Transport Mode' parameter to **Bypass** [FaxTransportMode = 2].
   - Configure the 'V.xx Modem Transport Type' parameters to **Enable Bypass** [VxxModemTransportType = 2].
2. Configure the [FaxModemNTEMode] parameter to [1] (enables this feature).

V.34 Fax Support

V.34 fax machines can transmit data over IP to the remote side using various methods. The device supports the following modes for transporting V.34 fax data over IP:

- Bypass mechanism for V.34 fax transmission (see Bypass Mechanism for V.34 Fax Transmission)
- T.38 Version 0 relay mode, i.e., fallback to T.38 (see Relay Mode for T.30 and V.34 Faxes)

To configure whether to pass V.34 over T.38 fax relay, or use Bypass over the High Bit Rate coder (e.g. PCM A-Law), use the 'V.34 Fax Transport Type' parameter (V34FaxTransportType).

You can use the 'SIP T.38 Version' parameter (SIPT38Version) to configure:

- Pass V.34 over T.38 fax relay using bit rates of up to 33,600 bps ('SIP T.38 Version' is set to Version 3).
- Use Fax-over-T.38 fallback to T.30, using up to 14,400 bps ('SIP T.38 Version' is set to Version 0).

- The CNG detector is disabled in all the subsequent examples. To disable the CNG detector, set the 'CNG Detector Mode' parameter (CNGDetectorMode) to **Disable**.

Bypass Mechanism for V.34 Fax Transmission

In this proprietary scenario, the device uses bypass (or NSE) mode to transmit V.34 faxes, enabling the full utilization of its speed.

➢ **To use bypass mode for T.30 and V.34 faxes:**

1. On the Fax/Modem/CID Settings page, do the following:
   - Set the 'Fax Transport Mode' parameter to **Bypass** [FaxTransportMode = 2].
   - Set the 'V.22 Modem Transport Type' parameter to **Enable Bypass** [V22ModemTransportType = 2].
   - Set the 'V.23 Modem Transport Type' parameter to **Enable Bypass** [V23ModemTransportType = 2].
   - Set the 'V.32 Modem Transport Type' parameter to **Enable Bypass** [V32ModemTransportType = 2].
   - Set the 'V.34 Modem Transport Type' parameter to **Enable Bypass** [V34ModemTransportType = 2].
2. Configure the [V34FaxTransportType] parameter to [2] (Bypass).

➢ **To use bypass mode for V.34 faxes, and T.38 for T.30 faxes:**

1. On the Fax/Modem/CID Settings page, do the following:
   - Set the 'Fax Transport Mode' parameter to **T.38 Relay** [FaxTransportMode = 1].
   - Set the 'V.22 Modem Transport Type' parameter to **Enable Bypass** [V22ModemTransportType = 2].
   - Set the 'V.23 Modem Transport Type' parameter to **Enable Bypass** [V23ModemTransportType = 2].
   - Set the 'V.32 Modem Transport Type' parameter to **Enable Bypass** [V32ModemTransportType = 2].
   - Set the 'V.34 Modem Transport Type' parameter to **Enable Bypass** [V34ModemTransportType = 2].
2. Configure the [V34FaxTransportType] parameter to [2] (Bypass).

Relay Mode for T.30 and V.34 Faxes

In this scenario, V.34 fax machines are forced to use their backward compatibility with T.30 faxes and operate in the slower T.30 mode.

➢ **To use T.38 mode for V.34 and T.30 faxes:**

1. On the Fax/Modem/CID Settings page, do the following:
   - Set the 'Fax Transport Mode' parameter to **T.38 Relay** (FaxTransportMode = 1).
   - Set the 'V.22 Modem Transport Type' parameter to **Disable** (V22ModemTransportType = 0).
   - Set the 'V.23 Modem Transport Type' parameter to **Disable** (V23ModemTransportType = 0).
   - Set the 'V.32 Modem Transport Type' parameter to **Disable** (V32ModemTransportType = 0).
   - Set the 'V.34 Modem Transport Type' parameter to **Disable** (V34ModemTransportType = 0).
2. Configure the [V34FaxTransportType] parameter to [1] (Relay).

➢ **To allow V.34 fax relay over T.38:**

- Set the 'SIP T.38 Version' parameter to Version 3 (SIPT38Version = 3).

➢ **To force V.34 fax machines to use their backward compatibility with T.30 faxes and operate in the slower T.30 mode:**

- Set the 'SIP T.38 Version' parameter to Version 0 (SIPT38Version = 0).

For SBC calls, the device forwards T.38 Version 3 transparently (as is) to the other leg (i.e., no transcoding).

V.34 Fax Relay for SG3 Fax Machines

Super Group 3 (SG3) is a standard for fax machines that support speeds of up to 33.6 kbps through V.34 half duplex (HD) modulation. The following procedure describes how to configure V.34 (SG3) fax relay support based on ITU Specification T.38 Version 3.

➢ **To enable support for V.34 fax relay (T.38) at SG3 speed:**

1. In the IP Profiles table (see Configuring IP Profiles), configure an IP Profile with the 'Fax Signaling Method' parameter (IpProfile_IsFaxUsed) set to **T.38 Relay**.
2. In the Coder Groups table (see Configuring Coder Groups) set the coder used by the device to G.729 (or any other supported codec).

3. On the Fax/Modem/CID Settings page, do the following settings:
   - 'SIP T.38 Version' to **Version 3** (SIPT38Version = 3).
   - 'Fax Relay Max Rate' (RelayMaxRate) to **33,600bps** (default).
   - 'CNG Detector Mode' (CNGDetectorMode) to **Disable** (default).
   - 'V.21 Modem Transport Type' to **Disable** (V21ModemTransportType = 0).
   - 'V.22 Modem Transport Type' to **Disable** (V22ModemTransportType = 0).
   - 'V.23 Modem Transport Type' to **Disable** (V23ModemTransportType = 0).
   - 'V.32 Modem Transport Type' to **Disable** (V32ModemTransportType = 0).
   - 'V.34 Modem Transport Type' to **Disable** (V34ModemTransportType = 0).
   - 'CED Transfer Mode' to Fax Relay or VBD (CEDTransferMode = 0).
4. Set the ini file parameter, V34FaxTransportType to 1 (i.e., relay).
5. Set the ini file parameter, T38MaxDatagramSize to 560 (default).

- The T.38 negotiation should be completed at call start according to V.152 procedure (as shown in the INVITE example below).
- T.38 mid-call Re-INVITEs are supported.
- If the remote party supports only T.38 Version 0, the device "downgrades" the T.38 Version 3 to T.38 Version 0.

For example, the device sends or receives the following INVITE message, negotiating both audio and image media:

```
INVITE sip:2001@10.8.211.250;user=phone SIP/2.0
Via: SIP/2.0/UDP 10.8.6.55;branch=z9hG4bKac1938966220
Max-Forwards: 70
From: <sip:318@10.8.6.55>;tag=1c1938956155
To: <sip:2001@10.8.211.250;user=phone>
Call-ID: 193895529241200022331@10.8.6.55
CSeq: 1 INVITE
Contact: <sip:318@10.8.6.55:5060>
Supported: em,100rel,timer,replaces,path,resource-priority,sdp-anat
Allow:
REGISTER,OPTIONS,INVITE,ACK,CANCEL,BYE,NOTIFY,PRACK,REFER,INF
O,SUBSCRIBE,UPDATE
Remote-Party-ID:
<sip:318@10.8.211.250>;party=calling;privacy=off;screen=no;screen-
ind=0;npi=1;ton=0
Remote-Party-ID: <sip:2001@10.8.211.250>;party=called;npi=1;ton=0
User-Agent: AudioCodes-Sip-Gateway-/7.20A.258.980
```

```
Content-Type: application/sdp
Content-Length: 433

v=0
o=AudioCodesGW 1938931006 1938930708 IN IP4 10.8.6.55
s=Phone-Call
c=IN IP4 10.8.6.55
t=0 0
m=audio 6010 RTP/AVP 18 97
a=rtpmap:18 G729/8000
a=fmtp:18 annexb=no
a=rtpmap:97 telephone-event/8000
a=fmtp:97 0-15
a=ptime:20
a=sendrecv
m=image 6012 udptl t38
a=T38FaxVersion:3
a=T38MaxBitRate:33600
a=T38FaxMaxBuffer:1024
a=T38FaxMaxDatagram:122
a=T38FaxRateManagement:transferredTCF
a=T38FaxUdpEC:t38UDPRedundancy
```

V.152 Support

The device supports the ITU-T recommendation V.152 (Procedures for Supporting Voice-Band Data over IP Networks). Voice-band data (VBD) is the transport of modem, facsimile, and text telephony signals over a voice channel of a packet network with a codec appropriate for such signals.

For V.152 capability, the device supports T.38 as well as VBD codecs (i.e., G.711 A-law and G.711 µ-law). The selection of capabilities is performed using the Coder Groups table (see Configuring Coder Groups).

When in VBD mode for V.152 implementation, support is negotiated between the device and the remote endpoint at the establishment of the call. During this time, initial exchange of call capabilities is exchanged in the outgoing SDP. These capabilities include whether VBD is supported and associated RTP payload types ('gpmd' SDP attribute), supported codecs, and packetization periods for all codec payload types ('ptime' SDP attribute). After this initial negotiation, no Re-INVITE messages are necessary as both endpoints are synchronized in terms of the other side's capabilities. If negotiation fails (i.e., no match was achieved for any of the transport capabilities), fallback to existing logic occurs (according to the parameter IsFaxUsed).

Below is an example of media descriptions of an SDP indicating support for V.152. In the example, V.152 implementation is supported (using the dynamic payload type 96 and G.711 u-law as the VBD codec) as well as the voice codecs G.711 µ-law and G.729.

```
v=0
o=-  0 0 IN IPV4 <IPAdressA>
s=-
t=0 0
p=+1
c=IN IP4  <IPAddressA
m=audio <udpPort A> RTP/AVP 18 0
a=ptime:10
a=rtpmap:96 PCMU/8000
a=gpmd: 96 vbd=yes
```

Instead of using VBD transport mode, the V.152 implementation can use alternative relay fax transport methods (e.g., fax relay over IP using T.38). The preferred V.152 transport method is indicated by the SDP 'pmft' attribute. Omission of this attribute in the SDP content means that VBD mode is the preferred transport mechanism for voice-band data.

Configuring RTP/RTCP Settings

This section describes configuration relating to Real-Time Transport Protocol (RTP) and RTP Control Protocol (RTCP).

Configuring the Dynamic Jitter Buffer

Voice frames are transmitted at a fixed rate. If the frames arrive at the other end at the same rate, voice quality is perceived as good. However, some frames may arrive slightly faster or slower than the other frames. This is called jitter (delay variation) and degrades the perceived voice quality. To minimize this problem, the device uses a jitter buffer. The jitter buffer collects voice packets, stores them and sends them to the voice processor in evenly spaced intervals.

The device uses a dynamic jitter buffer that can be configured with the following:

- **Minimum delay:** Defines the starting jitter capacity of the buffer. For example, at 0 msec, there is no buffering at the start. At the default level of 10 msec, the device always buffers incoming packets by at least 10 msec worth of voice frames.
- **Optimization Factor:** Defines how the jitter buffer tracks to changing network conditions. When set at its maximum value of 12, the dynamic buffer aggressively tracks changes in delay (based on packet loss statistics) to increase the size of the buffer and doesn't decay back down. This results in the best packet error performance, but at the cost of extra delay. At the minimum value of 0, the buffer tracks delays only to compensate for clock drift and quickly decays back to the minimum level. This optimizes the delay performance but at the expense of a higher error rate.

The default settings of 10 msec Minimum delay and 10 Optimization Factor should provide a good compromise between delay and error rate. The jitter buffer 'holds' incoming packets for 10 msec before making them available for decoding into voice. The coder polls frames from the buffer at regular intervals in order to produce continuous speech. As long as delays in the

network do not change (jitter) by more than 10 msec from one packet to the next, there is always a sample in the buffer for the coder to use. If there is more than 10 msec of delay at any time during the call, the packet arrives too late. The coder tries to access a frame and is not able to find one. The coder must produce a voice sample even if a frame is not available. It therefore compensates for the missing packet by adding a Bad-Frame-Interpolation (BFI) packet. This loss is then flagged as the buffer being too small. The dynamic algorithm then causes the size of the buffer to increase for the next voice session. The size of the buffer may decrease again if the device notices that the buffer is not filling up as much as expected. At no time does the buffer decrease to less than the minimum size configured by the Minimum delay parameter.

In certain scenarios, the **Optimization Factor is set to 13**: One of the purposes of the Jitter Buffer mechanism is to compensate for clock drift. If the two sides of the VoIP call are not synchronized to the same clock source, one RTP source generates packets at a lower rate, causing under-runs at the remote Jitter Buffer. In normal operation (optimization factor 0 to 12), the Jitter Buffer mechanism detects and compensates for the clock drift by occasionally dropping a voice packet or by adding a BFI packet.

Fax and modem devices are sensitive to small packet losses or to added BFI packets. Therefore, to achieve better performance during modem and fax calls, the Optimization Factor should be set to 13. In this special mode the clock drift correction is performed less frequently - only when the Jitter Buffer is completely empty or completely full. When such condition occurs, the correction is performed by dropping several voice packets simultaneously or by adding several BFI packets simultaneously, so that the Jitter Buffer returns to its normal condition.

The following procedure describes how to configure the jitter buffer using the Web interface.

➢ **To configure jitter buffer using the Web interface:**

1. Open the RTP/RTCP Settings page (**Setup** menu > **Signaling & Media** menu > **Media** folder > **RTP/RTCP Settings**). The relevant parameters are listed under the General group, as shown below:

2. Set the 'Dynamic Jitter Buffer Minimum Delay' parameter (DJBufMinDelay) to the minimum delay (in msec) for the Dynamic Jitter Buffer.
3. Set the 'Dynamic Jitter Buffer Optimization Factor' parameter (DJBufOptFactor) to the Dynamic Jitter Buffer frame error/delay optimization factor.
4. Click **Apply**.

Configuring RFC 2833 Payload

You can configure the RFC 2833 payload.

➢ **To configure RFC 2833 payload:**

1. Open the RTP/RTCP Settings page (**Setup** menu > **Signaling & Media** tab > **Media** folder > **RTP/RTCP Settings**).
2. Configure the following parameters:
   - 'RTP Redundancy Depth' (RTPRedundancyDepth) - enables the device to generate RFC 2198 redundant packets.
   - 'RFC 2833 TX Payload Type' (RFC2833TxPayloadType) - defines the Tx RFC 2833 DTMF relay dynamic payload type.
   - 'RFC 2833 RX Payload Type' (RFC2833RxPayloadType) - defines the Rx RFC 2833 DTMF relay dynamic payload type.
   - 'RFC 2198 Payload Type' (RFC2198PayloadType) - defines the RTP redundancy packet payload type according to RFC 2198.
3. Click **Apply**.

Configuring RTP Base UDP Port

You can configure the range (pool) of local UDP ports from which the device allocates ports to media (RTP, RTCP, and T.38) channels (legs). The range limit of UDP ports is from 6,000 through to 65,535.

The consecutive port offset from the RTP port for RTCP and T.38 traffic is one and two, respectively. For example, if the voice session uses RTP port 6000, the device allocates ports 6001 and 6002 for RTCP and T.38, respectively. However, you can configure the device to use the same port for RTP and T.38 packets, by configuring the [T38UseRTPPort] parameter to 1.

Within the port range, the device allocates the UDP ports per media channel (leg) in "jumps" (spacing) of 4 (default), 5, or 10, configured by the [UdpPortSpacing] parameter. For example, if the port range starts at 6000 and the UDP port spacing is 10, the available ports are 6000, 6010, 6020, 6030, and so on. Within the port range, the device assigns these ports **randomly** to the different media channels. For example, it allocates port 6000 to leg 1, port 6030 to leg 2, and port 6010 to leg 3.

You can configure the starting port (lower boundary) of the port range (default is 6000), using the 'RTP Base UDP Port' [BaseUDPPort] parameter. Once configured, the port range is according to the following equation:

```
<'RTP Base UDP Port' parameter value> to 65,535
```

For example: If you configure the 'RTP Base UDP Port' parameter to 6000, the port range is 6000 to 65,535.

You can also configure specific port ranges for specific SIP user agents (UAs), using Media Realms (see Configuring Media Realms). You can configure each Media Realm with a different

UDP port range and then assign the Media Realm to a specific IP Group, for example. However, the port range of the Media Realm **must be within the range** configured by the 'RTP Base UDP Port' parameter.

The following procedure describes how to configure the RTP base UDP port through the Web interface.

➢ **To configure the RTP base UDP port:**

1. Open the RTP/RTCP Settings page (**Setup** menu > **Signaling & Media** tab > **Media** folder > **RTP/RTCP Settings**).
2. In the 'RTP Base UDP Port' field, configure the lower boundary of the UDP port range.

3. Click **Apply**, and then reset the device with a save-to-flash for your settings to take effect.

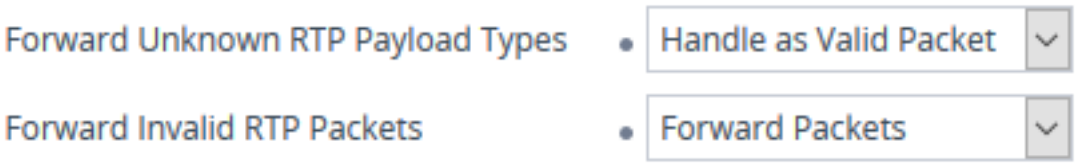
- The RTP port must be different from ports configured for SIP signaling traffic (i.e., ports configured for SIP Interfaces). For example, if the RTP port range is 6000 to 6999, the SIP port must either be less than 6000 or greater than 6999.

Configuring Invalid RTP/RTCP Packet Handling

You can configure the way the device handles incoming invalid RTP and RTCP packets. This is applicable only if you configure the IP Profile parameter, 'Mediation Mode' (IpProfile_TranscodingMode) to **RTP Forwarding**.

➢ **To configure invalid packet handling:**

1. Open the RTP/RTCP Settings page (**Setup** menu > **Signaling & Media** tab > **Media** folder > **RTP/RTCP Settings**).
2. From the 'Forward Unknown RTP Payload Types' drop-down list, select the required handling for RTP packets with unknown payload types.
3. From the 'Forward Invalid RTP Packets' drop-down list, select the required handling for invalid RTP and RTCP packets.



4. Click **Apply**.

Event Detection and Notification using X-Detect Header

The device can detect certain events in the media stream and notify of their detection to a remote application server, using the SIP X-Detect header. The request for event notification is done by the application server when establishing a SIP dialog (i.e., INVITE message) or during an already established call using a re-INVITE message. The device can detect the following event types:

- Answering Machine Detection (AMD): Detects events that are related to the AMD feature. AMD detects whether an answering machine or live voice has answered the call. It can also be used to detect silence, or the beep sound played by an answering machine to indicate the end of the greeting message after which a voice message can be left. For more information on AMD, see Answering Machine Detection (AMD).
- Call Progress Tone (CPT): Detects whether a specific tone, defined in the installed CPT file is received from the call. It can be used to detect the beep sound played by an answering machine (as mentioned above), and the busy, reorder and ring tones.

- Event detection on SBC calls for CPT is supported only for calls using the G.711 coder.

The X-Detect header is used for event detection as follows:

- X-Detect header in the INVITE message received from the application server requesting a specific event detection:

```
X-Detect: Request=[event type to detect]
```

- X-Detect header in the SIP response message -- SIP 183 (for early dialogs) or 200 OK (for confirmed dialogs) -- sent by the device to the application server specifying which of the requested events it can detect (absence of the X-Detect header indicates that the device cannot detect any of the events):

```
X-Detect: Response=[supported event types]
```

- Each time the device detects the supported event, it sends an INFO message to the remote party with the following message body:

```
Content-Type: Application/X-Detect
Type = [event type]
Subtype = [subtype of each event type]
```

The table below lists the event types and subtypes that can be detected by the device. The text shown in the table are the strings used in the X-Detect header. The table also provides a summary of the required configuration.

For SBC calls, event detection is enabled using the IPProfile_SBCHandleXDetect parameter in the IP Profiles table (see Configuring IP Profiles).

Table 15-1: Supported X-Detect Event Types

Event Type	Subtype	Description and Required Configuration
AMD	■ Voice (live voice) ■ Automata (answering machine) ■ Silence (no voice) ■ Unknown ■ Beep (greeting message of answering machine)	Event detection using the AMD feature. For more information, see Answering Machine Detection (AMD).
CPT	■ Busy ■ Reorder ■ Ringtone ■ Beep (greeting message of answering message)	Event detection of tones using the CPT file. 1. Create a CPT file with the required tone types of the events that you want to detect. 2. Install the CPT file on the device. **Note:** ■ To configure beep detection, see Detecting Answering Machine Beep.
FAX	CED	■ Set the IsFaxUsed parameter to any value other than 0.
PTT	■ voice-start ■ voice-end	Set the EnableDSPIPMDetectors parameter to 1.

Detecting Answering Machine Beeps

The device can detect the "beep" sound played by an answering machine that indicates the end of the answering machine's greeting message. This is useful in that the device can then notify, for example, a third-party, application server that it can now leave a voice message on the answering machine. The device supports the following methods for detecting and reporting beeps:

- **AMD-based Detection:** The device uses its beep detector that is integrated in the AMD feature. You can configure the beep detection timeout and beep detection sensitivity level

(for more information, see Configuring AMD). To enable the AMD beep detection, the received INVITE message must contain an X-Detect header with the value "Request=AMD",

```
X-Detect: Request=AMD
```

and the [AMDBeepDetectionMode] parameter must be configured to [1] or [2]. If configured to [1], the beep is detected only after the answering machine is detected. If configured to [2], the beep is detected even if the answering machine was not detected.

- **Tone-based Detection (Call Progress Tone):** The device detects the beep according to a call progress tone (CPT). This is enabled if the device receives a specific beep tone (Tone Type #46) that is also defined in the installed CPT file and the received INVITE message contains an X-Detect header with the value "Request=CPT":

  ```
  X-Detect: Request=CPT
  ```

  For more information on the CPT file, see Call Progress Tones File.

The device reports beep detections to application servers, by sending a SIP INFO message that contains a body with one of the following values, depending on the method used for detecting the beep:

- AMD-detected Beep:

  ```
  Type= AMD
  SubType= Beep
  ```

- CPT-detected Beep:

  ```
  Type= CPT
  SubType=Beep
  ```

SIP Call Flow Examples of Event Detection and Notification

Two SIP call flow examples are provided below of event detection and notification:

- **Example 1:** This example shows a SIP call flow of the device's AMD and event detection feature, whereby the device detects an answering machine and the subsequent start and end of the greeting message, enabling the third-party application server to know when to play a recorded voice message to an answering machine:

  a. Upon detection of the answering machine, the device sends the following SIP INFO message to the application server:

  ```
  INFO sip:sipp@172.22.2.9:5060 SIP/2.0
  Via: SIP/2.0/UDP 172.22.168.249;branch=z9hG4bKac1566945480
  Max-Forwards: 70
  ```

```
From: sut <sip:3000@172.22.168.249:5060>;tag=1c1505895240
To: sipp <sip:sipp@172.22.2.9:5060>;tag=1
Call-ID: 1-29758@172.22.2.9
CSeq: 1 INFO
Contact: <sip:56700@172.22.168.249>
Supported: em,timer,replaces,path,resource-priority
Allow:
REGISTER,OPTIONS,INVITE,ACK,CANCEL,BYE,NOTIFY,PRACK,REFER,INFO,SUBSCRIBE,UPDATE
User-Agent: AudioCodes-Sip-Gateway/7.20A.258.980
Content-Type: application/x-detect
Content-Length: 30
Type= AMD
SubType= AUTOMATA
```

b. Upon detection of the start of voice (i.e., the greeting message of the answering machine), the device sends the following INFO message to the application server:

```
INFO sip:sipp@172.22.2.9:5060 SIP/2.0
Via: SIP/2.0/UDP 172.22.168.249;branch=z9hG4bKac482466515
Max-Forwards: 70
From: sut <sip:3000@172.22.168.249:5060>;tag=1c419779142
To: sipp <sip:sipp@172.22.2.9:5060>;tag=1
Call-ID: 1-29753@172.22.2.9
CSeq: 1 INFO
Contact: <sip:56700@172.22.168.249>
Supported: em,timer,replaces,path,resource-priority
Allow:
REGISTER,OPTIONS,INVITE,ACK,CANCEL,BYE,NOTIFY,PRACK,REFER,INFO,SUBSCRIBE,UPDATE
User-Agent: AudioCodes-Sip-Gateway/7.20A.258.980
Content-Type: application/x-detect
Content-Length: 34
Type= PTT
SubType= SPEECH-START
```

c. Upon detection of the end of voice (i.e., end of the greeting message of the answering machine), the device sends the following INFO message to the application server:

```
INFO sip:sipp@172.22.2.9:5060 SIP/2.0
Via: SIP/2.0/UDP 172.22.168.249;branch=z9hG4bKac482466515
Max-Forwards: 70
From: sut <sip:3000@172.22.168.249:5060>;tag=1c419779142
To: sipp <sip:sipp@172.22.2.9:5060>;tag=1
Call-ID: 1-29753@172.22.2.9
```

```
CSeq: 1 INFO
Contact: <sip:56700@172.22.168.249>
Supported: em,timer,replaces,path,resource-priority
Allow:
REGISTER,OPTIONS,INVITE,ACK,CANCEL,BYE,NOTIFY,PRACK,REFER,INFO,SUBSCRIBE,UPDATE
User-Agent: AudioCodes-Sip-Gateway/7.20A.258.980
Content-Type: application/x-detect
Content-Length: 34
Type= PTT
SubType= SPEECH-END
```

d. The application server sends its message to leave on the answering message.

- **Example 2:** This example shows a SIP call flow for event detection and notification of the beep of an answering machine:

a. The device receives a SIP message containing the X-Detect header from the remote application requesting beep detection:

```
INVITE sip:101@10.33.2.53;user=phone SIP/2.0
Via: SIP/2.0/UDP 10.33.2.53;branch=z9hG4bKac5906
Max-Forwards: 70
From: "anonymous" <sip:anonymous@anonymous.invalid>;tag=1c25298
To: <sip:101@10.33.2.53;user=phone>
Call-ID: 11923@10.33.2.53
CSeq: 1 INVITE
Contact: <sip:100@10.33.2.53>
X-Detect: Request=AMD,CPT
```

b. The device sends a SIP response message to the remote party, listing the events in the X-Detect header that it can detect:

```
SIP/2.0 200 OK
Via: SIP/2.0/UDP 10.33.2.53;branch=z9hG4bKac5906
From: "anonymous" <sip:anonymous@anonymous.invalid>;tag=1c25298
To: <sip:101@10.33.2.53;user=phone>;tag=1c19282
Call-ID: 11923@10.33.2.53
CSeq: 1 INVITE
Contact: <sip:101@10.33.2.53>
X-Detect: Response=AMD,CPT
```

c. The device detects the beep of an answering machine and sends an INFO message to the remote party:

```
INFO sip:101@10.33.2.53;user=phone SIP/2.0
Via: SIP/2.0/UDP 10.33.2.53;branch=z9hG4bKac5906
Max-Forwards: 70
From: "anonymous" <sip:anonymous@anonymous.invalid>;tag=1c25298
To: <sip:101@10.33.2.53;user=phone>
Call-ID: 11923@10.33.2.53
CSeq: 1 INVITE
Contact: <sip:100@10.33.2.53>
X- Detect: Response=AMD,CPT
Content-Type: Application/X-Detect
Content-Length: xxx
Type = CPT
Subtype = Beep
```

Answering Machine Detection (AMD)

The device's Answering Machine Detection (AMD) feature can detect whether an outbound call has been answered by a human (including fax) or an answering machine. The device analyzes the sound (speech) patterns received in the first few seconds of the call to determine whether a human (live person) or machine has answered the call. Typically, when a human answers the call, there is a short "hello ..." followed by silence to wait for the other party to respond. In contrast, when an answering machine answers the call, there is constant speech (answering message) followed by a beep to leave a voice-mail message.

When the device detects what answered the call (human or machine), it can notify this detection type to, for example, a third-party application server used for automatic dialing applications. The X-Detect SIP header is used for requesting event detection and notification. For more information, see Event Detection and Notification using X-Detect Header. The device can also detect beeps played by an answering machine at the end of its greeting message. For more information, see Detecting Answering Machine Beeps.

The device's default AMD feature is based on voice detection for North American English (see note below). It uses sophisticated speech detection algorithms which are based on hundreds of real-life recordings of answered calls by live voice and answering machines in English. The algorithms are used to detect whether it's human or machine based on voice and silence duration as well as speech patterns. The algorithms of the language-based recordings are compiled into a file called AMD Sensitivity. This file is provided by default, pre-installed on the device.

As the main factor (algorithm) for detecting human and machine is the voice pattern and silence duration, the language on which the detection algorithm is based, is in most cases not important as these factors are similar across most languages. Therefore, the default, pre-installed AMD Sensitivity file, which is based on North American English, may suffice your deployment even if the device is located in a region where a language other than English is used.

However, if (despite the information stated in the note above) you wish to implement AMD in a different language or region, or if you wish to fine-tune the default AMD algorithms to suit your specific deployment, please contact the sales representative of your purchased device for more information on this service. You will be typically required to provide AudioCodes with a database of recorded voices (calls) in the language on which the device's AMD feature can base its voice detector algorithms. The data needed for an accurate calibration should be recorded under the following guidelines:

- Statistical accuracy: The number of recorded calls should be as high as possible (at least 100) and varied. The calls must be made to different people. The calls must be made in the specific location in which the device's AMD feature is to operate.
- Real-life recording: The recordings should simulate real-life answering of a called person picking up the phone, and without the caller speaking.
- Normal environment interferences: The environment in which the recordings are done should simulate real-life scenarios, in other words, not sterile but not too noisy either. Interferences, for example, could include background noises of other people talking, spikes, and car noises.

Once you have provided AudioCodes with your database of recordings, AudioCodes compiles it into a loadable file. For a brief description of the file format and for installing the file on the device, see AMD Sensitivity File.

The device supports up to eight AMD algorithm suites called *Parameter Suites*, where each suite defines a range of detection sensitivity levels. Sensitivity levels refer to how accurately the device's voice detection algorithms can detect if a human or machine has answered the call. Each level supports a different detection sensitivity to human and machine. For example, a specific sensitivity level may be more sensitive to detecting human than machine. In deployments where the likelihood of a call answered by an answering machine is low, it would be advisable to configure the device to use a sensitivity level that is more sensitive to human than machine. In addition, this allows you to tweak your sensitivity to meet local regulatory rules designed to protect consumers from automatic dialers (where, for example, the consumer picks up the phone and hears silence). Each suite can support up to 16 sensitivity levels (0 to 15), except for Parameter Suite 0, which supports up to 8 levels (0 to 7). The default, pre-installed AMD Sensitivity file, based on North American English, provides the following Parameter Suites:

- Parameter Suite 0 (normal sensitivity) - contains 8 sensitivity detection levels
- Parameter Suite 1 (high sensitivity) - contains 16 sensitivity detection levels

As Parameter Suite 1 provides a greater range of detection sensitivity levels (i.e., higher detection resolution), this may be the preferable suite to use in your deployment. The detected AMD type (human or machine) and success of detecting it correctly are sent in CDR and Syslog messages. For more information, see Syslog Fields for Answering Machine Detection (AMD).

The Parameter Suite and sensitivity level can be applied globally for all calls, or for specific calls using IP Profiles. For enabling AMD and selecting the Parameter Suite and sensitivity level, see Configuring AMD.

The tables below show the success rates of the default, pre-installed AMD Sensitivity file (based on North American English) for correctly detecting "live" human voice and answering machine:

Table 15-2: Approximate AMD Normal Detection Sensitivity - Parameter Suite 0 (Based on North American English)

AMD Detection Sensitivity Level	Performance	
	Success Rate for Live Calls	Success Rate for Answering Machine
0 (Best for Answering Machine)	-	-
1	82.56%	97.10%
2	85.87%	96.43%
3	88.57%	94.76%
4	88.94%	94.31%
5	90.42%	91.64%
6	90.66%	91.30%
7 (Best for Live Calls)	94.72%	76.14%

Table 15-3: Approximate AMD High Detection Sensitivity - Parameter Suite 1 (Based on North American English)

AMD Detection Sensitivity Level	Performance	
	Success Rate for Live Calls	Success Rate for Answering Machine
0 (Best for Answering Machine)	72%	97%
1	77%	96%
2	79%	95%
3	80%	95%
4	84%	94%

AMD Detection Sensitivity Level	Performance	
5	86%	93%
6	87%	92%
7	88%	91%
8	90%	89%
9	90%	88%
10	91%	87%
11	94%	78%
12	94%	73%
13	95%	65%
14	96%	62%
15 (Best for Live Calls)	97%	46%

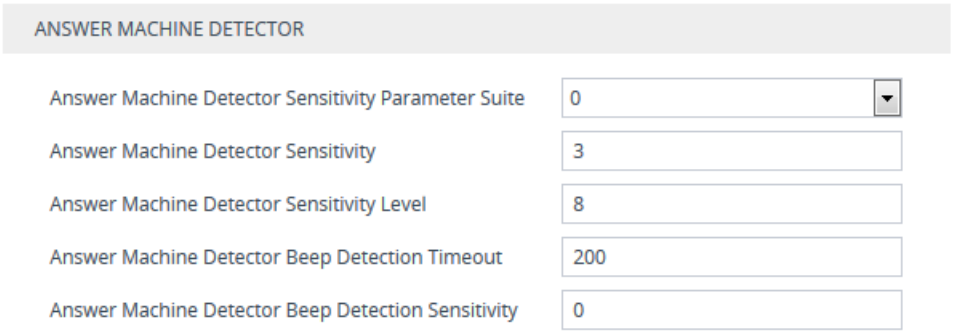
Configuring AMD

You can configure AMD for all calls using global AMD parameters or for specific calls using IP Profiles. The procedure below describes how to configure AMD for all calls. To configure AMD for specific calls, use the AMD parameters in the IP Profiles table (see Configuring IP Profiles)

➢ **To configure AMD for all calls:**

1. Open the DSP Settings page (**Setup** menu > **Signaling & Media** tab > **Media** folder > **DSP Settings**):
2. From the 'IPMedia Detectors' drop-down list (EnableDSPIPMDetectors), select **Enable** .

3. Scroll down to the Answer Machine Detector group:



4. Select the AMD algorithm suite:
   a. In the 'Answer Machine Detector Sensitivity Parameter Suite' field, select the required Parameter Suite included in the installed AMD Sensitivity file.
   b. In the 'Answer Machine Detector Sensitivity' field, enter the required detection sensitivity level of the selected Parameter Suite.
5. Configure the answering machine beep detection:
   a. In the 'Answer Machine Detector Beep Detection Timeout' field [AMDBeepDetectionTimeout], enter the duration that the beep detector operates from when detection is initiated.
   b. In the 'Answer Machine Detector Beep Detection Sensitivity' field [AMDBeepDetectionSensitivity], enter the AMD beep detection sensitivity level.
6. Click **Apply**, and then reset the device with a save-to-flash for your settings to take effect.

For a complete list of AMD-related parameters, see IP Media Parameters.

Automatic Gain Control (AGC)

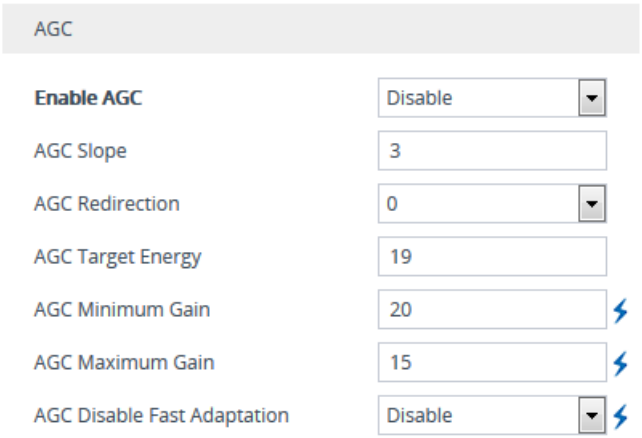
Automatic Gain Control (AGC) adjusts the energy of the output signal to a required level (volume). This feature compensates for near-far gain differences. AGC estimates the energy of the incoming signal from the IP side, determined by the 'AGC Redirection' parameter, calculates the essential gain, and then performs amplification. Feedback ensures that the output signal is not clipped. You can configure the required Gain Slope in decibels per second using the 'AGC Slope' parameter and the required signal energy threshold using the 'AGC Target Energy' parameter.

When the AGC first detects an incoming signal, it begins operating in Fast Mode, which allows the AGC to adapt quickly when a conversation starts. This means that the Gain Slope is 8 dB/sec for the first 1.5 seconds. After this period, the Gain Slope is changed to the user-defined value. You can disable or enable the AGC's Fast Mode feature, using the 'AGC Disable Fast Adaptation' parameter. After Fast Mode is used, the signal should be off for two minutes in order to have the feature turned on again.

➢ **To configure AGC:**

1. Open the DSP Settings page (**Setup** menu > **Signaling & Media** tab > **Media** folder > **DSP Settings**).
2. From the 'IPMedia Detectors' drop-down list [EnableDSPIPMDetectors], select **Enable**.

3. Configure the following AGC parameters:
   - 'Enable AGC' [EnableAGC] - enables the AGC mechanism.
   - 'AGC Slope' [AGCGainSlope] - defines the AGC convergence rate.
   - 'AGC Redirection' [AGCRedirection] - defines the AGC direction.
   - 'AGC Target Energy' - defines the signal energy value (dBm) that the AGC attempts to attain.
   - 'AGC Minimum Gain' [AGCMinGain] - defines the minimum gain (in dB) by the AGC when activated.
   - 'AGC Maximum Gain' [AGCMaxGain] - defines the maximum gain (in dB) by the AGC when activated.
   - 'AGC Disable Fast Adaptation' [AGCDisableFastAdaptation] - enables the AGC Fast Adaptation mode.



4. Configure the 'Transcoding Mode' [TranscodingMode] parameter to **Force**. You can configure this using the global parameter or per IP Profile.
5. Click **Apply**.

Configuring Media (SRTP) Security

The device supports Secured RTP (SRTP) according to RFC 3711. SRTP is used to encrypt RTP and RTCP transport for protecting VoIP traffic. SRTP requires a cryptographic key exchange mechanism to negotiate the keys. To negotiate the keys, the device supports the Session

Description Protocol Security Descriptions (SDES) protocol (according to RFC 4568), or Datagram Transport Layer Security (DTLS) protocol for SBC calls. For more information on DTLS, see SRTP using DTLS Protocol. The key exchange is done by adding the 'a=crypto' attribute to the SDP. This attribute is used (by both sides) to declare the various supported cipher suites and to attach the encryption key. If negotiation of the encryption data is successful, the call is established. Typically, 'a=crypto' is included in secured media (RTP/SAVP). However, there is also support for including 'a=crypto' in non-secured media (RTP/AVP). In such cases, the media is handled as if the device received two identical media: one secured and one not.

SRTP supports the following cipher suites (all other suites are ignored):

- AES_CM_128_HMAC_SHA1_32
- AES_CM_128_HMAC_SHA1_80
- AES_256_CM_HMAC_SHA1_32 (RFC 6188)
- AES_256_CM_HMAC_SHA1_80 (RFC 6188)

When the device is the offering side (SDP offer), it can generate a Master Key Identifier (MKI). You can configure the MKI size globally (by the [SRTPTxPacketMKISize] parameter) or per SIP entity (by the IP Profile parameter [IpProfile_MKISize]). The length of the MKI is limited to four bytes. If the remote side sends a longer MKI, the key is ignored.

- The device can forward MKI size transparently for SRTP-to-SRTP media flows or override the MKI size during negotiation (inbound or outbound leg).

The key lifetime field is not supported. However, if it is included in the key it is ignored and the call does not fail. For SBC calls belonging to a specific SIP entity, you can configure the device to remove the lifetime field in the 'a=crypto' attribute (by the IP Profile parameter [IpProfile_SBCRemoveCryptoLifetimeInSDP]).

For SDES, the keys are sent in the SDP body ('a=crypto') of the SIP message and are typically secured using SIP over TLS (SIPS). The encryption of the keys is in plain text in the SDP. The device supports the following session parameters:

- UNENCRYPTED_SRTP
- UNENCRYPTED_SRTCP
- UNAUTHENTICATED_SRTP

Session parameters should be the same for the local and remote sides. When the device is the offering side, the session parameters are configured by the following parameters - 'Authentication on Transmitted RTP Packets', 'Encryption on Transmitted RTP Packets, and 'Encryption on Transmitted RTCP Packets'. When the device is the answering side, the device adjusts these parameters according to the remote offering. Unsupported session parameters are ignored, and do not cause a call failure.

Below is an example of crypto attributes usage:

```
a=crypto:1 AES_CM_128_HMAC_SHA1_80
inline:PsKoMpHlCg+b5X0YLuSvNrImEh/dAe
a=crypto:2 AES_CM_128_HMAC_SHA1_32
inline:IsPtLoGkBf9a+c6XVzRuMqHlDnEiAd
```

The device also supports symmetric MKI negotiation, whereby it can forward the MKI size received in the SDP offer 'a=crypto' line in the SDP answer. You can enable symmetric MKI globally (by the [EnableSymmetricMKI] parameter) or per SIP entity (using the IP Profile parameter [IpProfile_EnableSymmetricMKI] and for SBC calls, [IpProfile_SBCEnforceMKISize]). For more information on symmetric MKI, see Configuring IP Profiles.

You can configure the enforcement policy of SRTP, by the [EnableMediaSecurity] parameter and [IpProfile_SBCMediaSecurityBehaviour] parameter for SBC calls. For example, if negotiation of the cipher suite fails or if incoming calls exclude encryption information, the device can be configured to reject the calls.

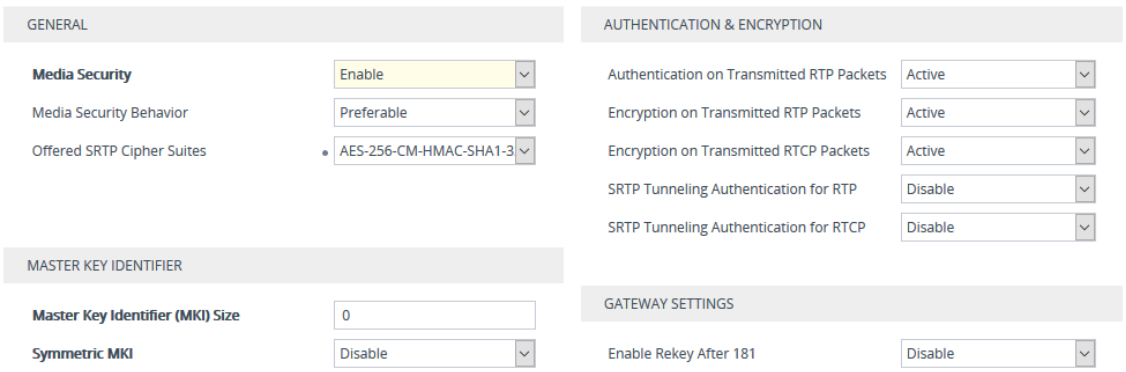
You can also enable the device to validate the authentication of packets for SRTP tunneling for RTP and RTCP. This applies only to SRTP-to-SRTP SBC calls and where the endpoints use the same key. This is configured using the 'SRTP Tunneling Authentication for RTP' and 'SRTP Tunneling Authentication for RTCP' parameters.

- For a detailed description of the SRTP parameters, see Configuring IP Profiles and SRTP Parameters.
- When SRTP is used, channel capacity may be reduced.

The procedure below describes how to configure SRTP through the Web interface.

➢ To enable and configure SRTP:

1. Open the Media Security page (**Setup** menu > **Signaling & Media** tab > **Media** folder > **Media Security**).



2. From the 'Media Security' drop-down list [EnableMediaSecurity], select **Enable** to enable SRTP.
3. From the 'Offered SRTP Cipher Suites' drop-down list [SRTPofferedSuites], select the supported cipher suite.

4. Configure the other SRTP parameters as required.
5. Click **Apply**.

SRTP using DTLS Protocol

For SBC calls, you can configure the device to use the Datagram Transport Layer Security (DTLS) protocol to secure UDP-based traffic (according to RFC 4347 and 6347) for specific SIP entities, using IP Profiles. DTLS allows datagram-based applications to communicate in a way that is designed to prevent eavesdropping, tampering or message forgery. The DTLS protocol is based on the stream-oriented TLS protocol, providing similar security. The device can therefore interwork in mixed environments where one network may require DTLS and the other may require Session Description Protocol Security Descriptions (SDES) or even non-secure RTP. The device supports DTLS negotiation for RTP-to-SRTP and SRTP-to-SRTP calls.

DTLS support is important for deployments with WebRTC. WebRTC requires that media channels be encrypted through DTLS for SRTP key exchange. Negotiation of SRTP keys through DTLS is done during the DTLS handshake between WebRTC client and peer. For more information on WebRTC, see WebRTC.

In contrast to SDES, DTLS key encryption is done over the media channel (UDP) and not over the signaling channel. Thus, DTLS-SRTP is generally known as "secured key exchange over media". DTLS is similar to TLS, but runs over UDP, whereas TLS is over TCP. Before the DTLS handshake, the peers exchange DTLS parameters (fingerprint and setup) and algorithm types in the SDP body of the SIP messages exchanged for establishing the call (INVITE request and response). The peers participate in a DTLS handshake during which they exchange certificates. These certificates are used to derive a symmetric key, which is used to encrypt data (SRTP) flow between the peers. A hash value calculated over the certificate is transported in the SDP using the 'a=fingerprint' attribute. At the end of the handshake, each side verifies that the certificate it received from the other side fits the fingerprint from the SDP. To indicate DTLS support, the SDP offer/answer of the SIP message uses the 'a=setup' attribute. The 'a=setup:actpass' attribute value is used in the SDP offer by the device. This indicates that the device is willing to be either a client ('act') or a server ('pass') in the handshake. The 'a=setup:active' attribute value is used in the SDP answer by the device. This means that the device wishes to be the client ('active') in the handshake.

```
a=setup:actpass
a=fingerprint: SHA-1
\4A:AD:B9:B1:3F:82:18:3B:54:02:12:DF:3E:5D:49:6B:19:E5:7C:AB
```

DTLS cipher suite reuses the TLS cipher suite. The DTLS handshake is done for every new call configured for DTLS. In other words, unlike TLS where the connection remains "open" for future calls, a new DTLS connection is required for every new call. Note that the entire authentication and key exchange for securing the media traffic is handled in the media path through DTLS. The signaling path is used only to verify the peers' certificate fingerprints. DTLS messages are multiplexed onto the same ports that are used for the media.

➢ **To configure DTLS:**

1. In the TLS Context table (see Configuring TLS Certificate Contexts), configure a TLS Context with the DTLS version [TLSContexts_DTLSVersion].
2. Open the IP Groups table (see Configuring IP Groups), and then for the IP Group associated with the SIP entity, assign it the TLS Context for DTLS, using the 'Media TLS Context' parameter [IPGroup_DTLSContext].
3. Open the IP Profiles table (see Configuring IP Profiles), and then for the IP Profile associated with the SIP entity, configure the following:
   - Configure the 'SBC Media Security Mode' parameter [IPProfile_SBCMediaSecurityBehavior] to **Secured** or **Both**.
   - Configure the 'Media Security Method' parameter [IPProfile_SBCMediaSecurityMethod] to **DTLS**.
   - Configure the 'RTCP Mux' parameter [IpProfile_SBCRTCPMux] to **Supported**. Multiplexing is required as the DTLS handshake is done for the port used for RTP and thus, RTCP and RTP must be multiplexed onto the same port.
   - Configure the ini file parameter [SbcDtlsMtu] (or CLI command `configure voip > sbc settings > sbc-dtls-mtu`) to define the maximum transmission unit (MTU) size for the DTLS handshake.
4. Configure the minimum interval that the device waits between transmission of DTLS packets in the same DTLS handshake, using the ini file parameter [DTLSTimeBetweenTransmissions].

- The 'Cipher Server' parameter must be configured to "ALL".
- The device does not support forwarding of DTLS transparently between endpoints.

16 Services

This section describes configuration for various supported services.

DHCP Server Functionality

The device can serve as a Dynamic Host Configuration Protocol (DHCP) server that assigns and manages IP addresses from a user-defined address pool for DHCP clients. The DHCP server can also be configured to supply additional information to the requesting client such as the IP address of the TFTP server, DNS server, NTP server, and default router (gateway). The DHCP server functionality complies with IETF RFC 2131 and RFC 2132.

The DHCP server can service up to 10,000 DHCP clients. The DHCP clients are typically IP phones that are connected to the device's LAN port.

The DHCP server is activated when you configure a valid entry in the DHCP Servers table (see Configuring the DHCP Server) and associate it with an active IP network interface listed in the IP Interfaces table. When an IP phone on the LAN requests an IP address, the DHCP server allocates one from the address pool. In scenarios of duplicated IP addresses on the LAN (i.e., an unauthorized network device using one of the IP addresses of the DHCP address pool), the DHCP server detects this condition using an Address Resolution Protocol (ARP) request and temporarily blacklists the duplicated address.

You can also configure the DHCP server to respond **only** to DHCPDiscover requests from DHCP clients that contain a specific value for Option 60 (Vendor Class Identification). For more information, see Configuring the Vendor Class Identifier.

Configuring the DHCP Server

The DHCP Servers table lets you configure the device's DHCP server (only one). The DHCP Server table configures the DHCP server implementation. This includes configuring the DHCP IP address pool from where IP addresses are allocated to requesting DHCP clients, as well as configuring other information such as IP addresses of the DNS server, NTP server, default router (gateway), and SIP proxy server. The DHCP server sends the information in DHCP Options. The table below lists the DHCP Options that the DHCP server sends to the DHCP client and which are configurable in the DHCP Servers table.

Table 16-1: Configurable DHCP Options in DHCP Servers Table

DHCP Option Code	DHCP Option Name
Option 53	DHCP Message Type
Option 54	DHCP Server Identifier
Option 51	IP Address Lease Time

DHCP Option Code	DHCP Option Name
Option 1	Subnet Mask
Option 3	Router
Option 6	Domain Name Server
Option 44	NetBIOS Name Server
Option 46	NetBIOS Node Type
Option 42	Network Time Protocol Server
Option 2	Time Offset
Option 66	TFTP Server Name
Option 67	Boot file Name
Option 120	SIP Server

Once you have configured the DHCP server, you can configure the following:

- DHCP Vendor Class Identifier names (DHCP Option 60) - see Configuring the Vendor Class Identifier
- Additional DHCP Options - see Configuring Additional DHCP Options
- Static IP addresses for DHCP clients - see Configuring Static IP Addresses for DHCP Clients

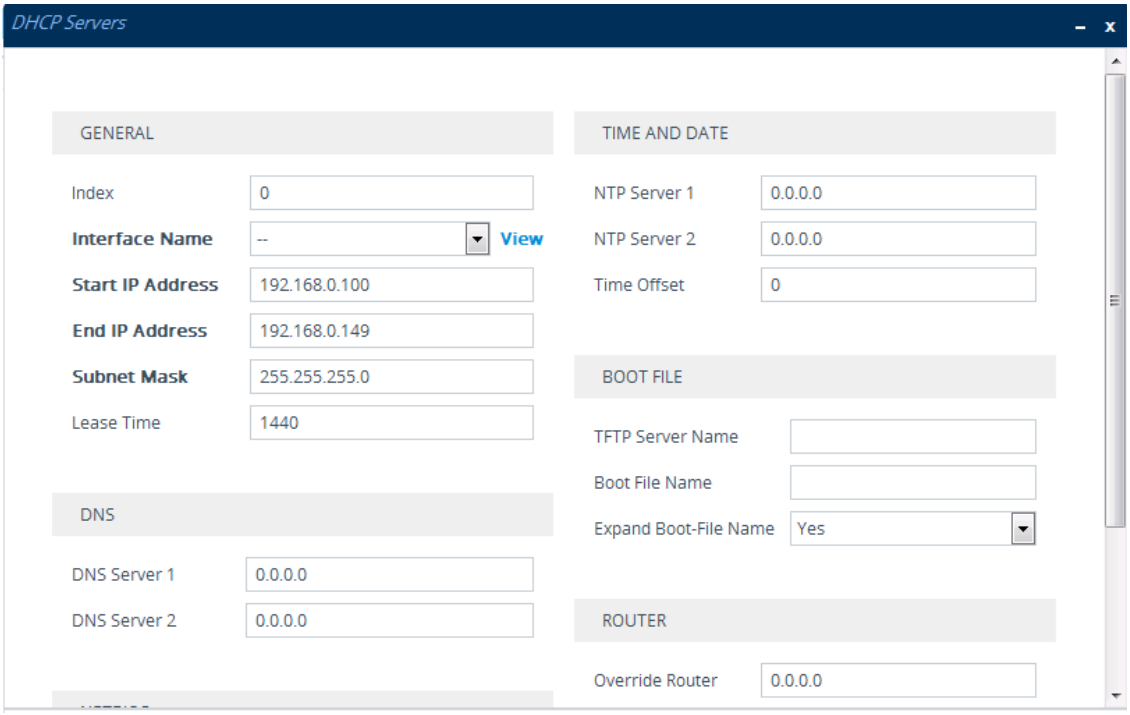
If you configure additional DHCP Options in the DHCP Option table, they override the default ones, which are configured in the DHCP Servers table. For example, if you configure Option 67 in the DHCP Option table, the device uses the value configured in the DHCP Option table instead of the value configured in the DHCP Servers table.

To view and delete currently serviced DHCP clients, see Viewing and Deleting DHCP Clients.

The following procedure describes how to configure the DHCP server through the Web interface. You can also configure it through ini file [DhcpServer] or CLI (`configure network > dhcp-server server <index>`).

➢ **To configure the device's DHCP server:**

1. Open the DHCP Servers page (**Setup** menu > **IP Network** tab > **Advanced** folder > **DHCP Severs**).
2. Click **New**; the following dialog box appears:



3. Configure a DHCP server according to the parameters described in the table below.
4. Click **Apply**.

Table 16-2: DHCP Servers Table Parameter Descriptions

Parameter	Description
General	
'Index' `dhcp server <index>`	Defines an index number for the new table row. **Note:** ■ Each row must be configured with a unique index. ■ Currently, only one index row can be configured.
'Interface Name' `network-if` [DhcpServer_ InterfaceName]	Associates an IP network interface on which the DHCP server operates. The IP interfaces are configured in the IP Interfaces table (see Configuring IP Network Interfaces). By default, no value is defined.
'Start IP Address' `start-address` [DhcpServer_ StartIPAddress]	Defines the starting IP address (IPv4 address in dotted-decimal format) of the IP address pool range used by the DHCP server to allocate addresses. The default value is 192.168.0.100. **Note:** The IP address must belong to the same subnet as the associated interface's IP address.

Parameter	Description
'End IP Address' end-address [DhcpServer_EndIPAddress]	Defines the ending IP address (IPv4 address in dotted-decimal format) of the IP address pool range used by the DHCP server to allocate addresses. The default value is 192.168.0.149. **Note:** The IP address must belong to the same subnet as the associated interface's IP address and must be "greater or equal" to the starting IP address defined in 'Start IP Address'.
'Subnet Mask' subnet-mask [DhcpServer_SubnetMask]	Defines the subnet mask (for IPv4 addresses) for the DHCP client. The value is sent in DHCP Option 1 (Subnet Mask). The default value is 0.0.0.0. **Note:** The value must be "narrower" or equal to the subnet mask of the associated interface's IP address. If set to "0.0.0.0", the subnet mask of the associated interface is used.
'Lease Time' lease-time [DhcpServer_LeaseTime]	Defines the duration (in minutes) of the lease time to a DHCP client for using an assigned IP address. The client needs to request a new address before this time expires. The value is sent in DHCP Option 51 (IP Address Lease Time). The valid value range is 0 to 214,7483,647. The default is 1440. When set to 0, the lease time is infinite.
DNS	
'DNS Server 1' dns-server-1 [DhcpServer_DNSServer1]	Defines the IP address (IPv4) of the primary DNS server that the DHCP server assigns to the DHCP client. The value is sent in DHCP Option 6 (Domain Name Server). The default value is 0.0.0.0.
'DNS Server 2' dns-server-2 [DhcpServer_DNSServer2]	Defines the IP address (IPv4) of the secondary DNS server that the DHCP server assigns to the DHCP client. The value is sent in DHCP Option 6 (Domain Name Server). The default value is 0.0.0.0.
NetBIOS	
'NetBIOS Name Server' netbios-server [DhcpServer_NetbiosNameServer]	Defines the IP address (IPv4) of the NetBIOS WINS server that is available to a Microsoft DHCP client. The value is sent in DHCP Option 44 (NetBIOS Name Server). The default value is 0.0.0.0.
'NetBIOS Node Type' netbios-node-type	Defines the node type of the NetBIOS WINS server for a Microsoft DHCP client. The value is sent in DHCP Option 46

Parameter	Description
[DhcpServer_ NetbiosNodeType]	(NetBIOS Node Type). ■ [0] **Broadcast** (default) ■ [1] **peer-to-peer** ■ [4] **Mixed** ■ [8] **Hybrid**
Time and Date	
'NTP Server 1' `ntp-server-1` [DhcpServer_ NTPServer1]	Defines the IP address (IPv4) of the primary NTP server that the DHCP server assigns to the DHCP client. The value is sent in DHCP Option 42 (Network Time Protocol Server). The default value is 0.0.0.0.
'NTP Server 2' `ntp-server-2` [DhcpServer_ NTPServer2]	Defines the IP address (IPv4) of the secondary NTP server that the DHCP server assigns to the DHCP client. The value is sent in DHCP Option 42 (Network Time Protocol Server). The default value is 0.0.0.0.
'Time Offset' `time-offset` [DhcpServer_ TimeOffset]	Defines the Greenwich Mean Time (GMT) offset (in seconds) that the DHCP server assigns to the DHCP client. The value is sent in DHCP Option 2 (Time Offset). The valid range is -43200 to 43200. The default is 0.
Boot File	
'TFTP Server Name' `tftp-server-name` [DhcpServer_TftpServer]	Defines the IP address or name of the TFTP server that the DHCP server assigns to the DHCP client. The TFTP server typically stores the boot file image, defined in the 'Boot file name' parameter (see below). The value is sent in DHCP Option 66 (TFTP Server Name). The valid value is a string of up to 80 characters. By default, no value is defined.
'Boot File Name' `boot-file-name` [DhcpServer_ BootFileName]	Defines the name of the boot file image for the DHCP client. The boot file stores the boot image for the client. The boot image is typically the operating system the client uses to load (downloaded from a boot server). The value is sent in DHCP Option 67 (Bootfile Name). To define the server storing the file, use the 'TFTP Server' parameter (see above). The valid value is a string of up to 256 characters. By default, no value is defined.

Parameter	Description
	The name can also include the following case-sensitive placeholder strings that are replaced with actual values if the 'Expand Boot-file Name' parameter is set to **Yes**: ■ <MAC>: Replaced by the MAC address of the client (e.g., *boot_<MAC>.ini*). The MAC address is obtained in the client's DHCP request. ■ <IP>: Replaced by the IP address assigned by the DHCP server to the client.
'Expand Boot-File Name' `expand-boot-file-name` [DhcpServer_ ExpandBootfileName]	Enables the use of the placeholders in the boot file name, defined in the 'Boot file name' parameter. ■ [0] **No** ■ [1] **Yes** (default)
Router	
'Override Router' `override-router-address` [DhcpServer_ OverrideRouter]	Defines the IP address (IPv4 in dotted-decimal notation) of the default router that the DHCP server assigns the DHCP client. The value is sent in DHCP Option 3 (Router). The default value is 0.0.0.0. If not specified (empty or "0.0.0.0"), the IP address of the default gateway configured in the IP Interfaces table for the IP network interface that you associated with the DHCP server (see the 'Interface Name' parameter above) is used.
SIP	
'SIP Server' `sip-server` [DhcpServer_SipServer]	Defines the IP address or DNS name of the SIP server that the DHCP server assigns the DHCP client. The client uses this SIP server for its outbound SIP requests. The value is sent in DHCP Option 120 (SIP Server). After defining the parameter, use the 'SIP server type' parameter (see below) to define the type of address (FQDN or IP address). The valid value is a string of up to 256 characters. The default is 0.0.0.0.
'SIP Server Type' `sip-server-type` [DhcpServer_ SipServerType]	Defines the type of SIP server address. The actual address is defined in the 'SIP server' parameter (see above). Encoding is done per SIP Server Type, as defined in RFC 3361. ■ [0] **DNS names** = (Default) The 'SIP server' parameter is configured with an FQDN of the SIP server.

Parameter	Description
	■ [1] **IP address** = The 'SIP server' parameter configured with an IP address of the SIP server.

Configuring the Vendor Class Identifier

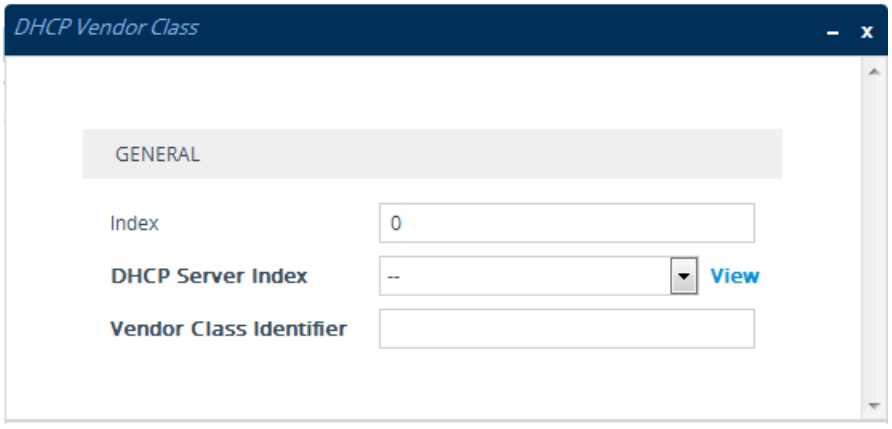
The DHCP Vendor Class table lets you configure up to 10 Vendor Class Identifier (VCI) names (DHCP Option 60). When the table is configured, the device's DHCP server responds only to DHCPDiscover requests that contain Option 60 and that match one of the DHCP VCIs configured in the table. If you have not configured any entries in the table, the DHCP server responds to all DHCPDiscover requests, regardless of the VCI.

The VCI is a string that identifies the vendor and functionality of a DHCP client to the DHCP server. For example, Option 60 can show the unique type of hardware (e.g., "AudioCodes 440HD IP Phone") or firmware of the DHCP client. The DHCP server can then differentiate between DHCP clients and process their requests accordingly.

The following procedure describes how to configure the DHCP VCIs through the Web interface. You can also configure it through ini file [DhcpVendorClass] or CLI (`configure network > dhcp-server vendor-class`).

➢ **To configure DHCP Vendor Class Identifiers:**

1. Open the DHCP Servers table (see Configuring the DHCP Server).
2. Select the row of the desired DHCP server for which you want to configure VCIs, and then click the **DHCP Vendor Class** link located below the table; the DHCP Vendor Class table opens.
3. Click **New**; the following dialog box appears:



4. Configure a VCI for the DHCP server according to the parameters described in the table below.
5. Click **Apply**.

Table 16-3: DHCP Vendor Class Table Parameter Descriptions

Parameter	Description
'Index' `dhcp vendor-class <index>` [DhcpVendorClass_Index]	Defines an index number for the new table row. **Note:** Each row must be configured with a unique index.
'DHCP Server Index' `dhcp-server-number` [DhcpVendorClass_DhcpServerIndex]	Associates the VCI table entry with a DHCP server that you configured in Configuring the DHCP Server. **Note:** Currently, only one DHCP server (Index 0) can be configured and therefore, the parameter is always set at 0.
'Vendor Class Identifier' `vendor-class` [DhcpVendorClass_VendorClassId]	Defines the value of the VCI DHCP Option 60. The valid value is a string of up to 80 characters. By default, no value is defined.

Configuring Additional DHCP Options

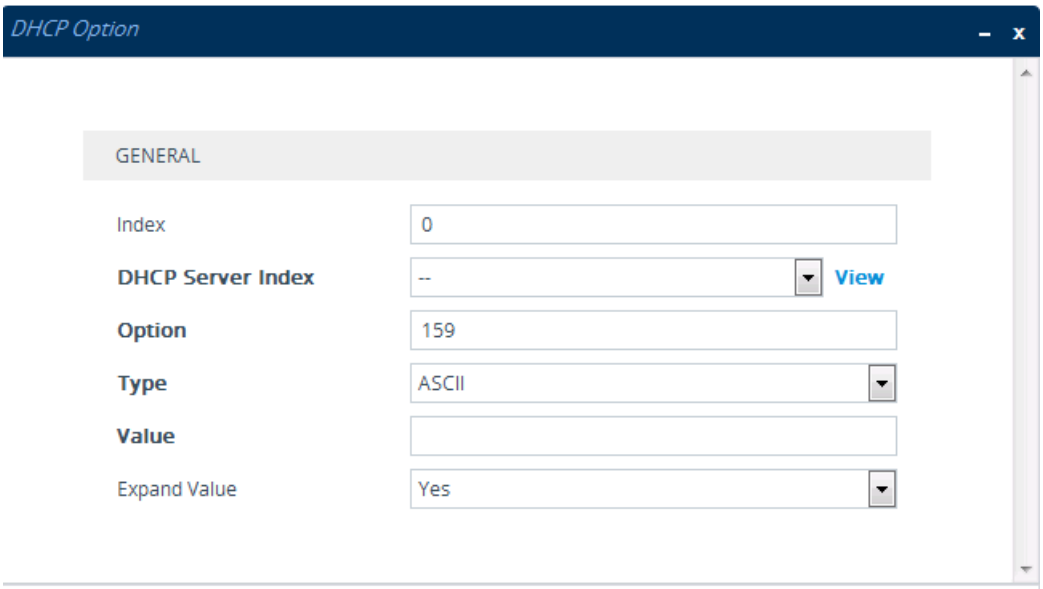
The DHCP Option table lets you configure up to 10 additional DHCP Options that the DHCP server can use to service the DHCP client. These DHCP Options are included in the DHCPOffer response sent by the DHCP server.

The following procedure describes how to configure DHCP Options through the Web interface. You can also configure it through ini file [DhcpOption] or CLI (`configure network > dhcp-server option`).

> **Note:** The additional DHCP Options configured in the DHCP Option table override the default ones, which are configured in the DHCP Servers table. In other words, if you configure Option 67 in the DHCP Option table, the device uses the value configured in the DHCP Option table instead of the value configured in the DHCP Servers table.

➢ **To configure DHCP Options:**

1. Open the DHCP Servers table (see Configuring the DHCP Server).
2. Select the row of the desired DHCP server for which you want to configure additional DHCP Options, and then click the **DHCP Option** link located below the table; the DHCP Option table opens.
3. Click **New**; the following dialog box appears:



4. Configure additional DHCP Options for the DHCP server according to the parameters described in the table below.
5. Click **Apply**.

Table 16-4: DHCP Option Table Parameter Descriptions

Parameter	Description
'Index' `dhcp option` [DhcpOption_Index]	Defines an index number for the new table row. **Note:** Each row must be configured with a unique index.
'DHCP Server Index' `dhcp-server-number` [DhcpOption_DhcpServerIndex]	Associates the DHCP Option table entry with a DHCP server that you configured in Configuring the DHCP Server. **Note:** Currently, only one DHCP server (Index 0) can be configured and therefore, the parameter is always set at 0.
'Option' `option` [DhcpOption_Option]	Defines the code of the DHCP Option. The valid value is 1 to 254. The default is 159. For example, for DHCP Option 150 (Cisco proprietary for defining multiple TFTP server IP addresses), enter the value 150.
'Type' `type` [DhcpOption_Type]	Defines the format (type) of the DHCP Option value that is configured in the 'Value' parameter (see below). ■ [0] **ASCII** = (Default) Plain-text string (e.g., when the value is a domain name).

Parameter	Description
	■ [1] **IP address** = IPv4 address. ■ [2] **Hexadecimal** = Hexadecimal-encoded string. For example, if you configure the 'Value' parameter to "company.com" (without quotation marks), you need to configure the 'Type' parameter to **ASCII**.
'Value' `value` [DhcpOption_Value]	Defines the value of the DHCP Option. For example, if you are using Option 66, the parameter is used for specifying the TFTP provisioning server (e.g., http://192.168.3.155:5000/provisioning/). The valid value is a string of up to 256 characters. By default, no value is defined. For IP addresses, the value can be one or more IPv4 addresses, each separated by a comma (e.g., 192.168.10.5,192.168.10.20). For hexadecimal values, the value is a hexadecimal string (e.g., c0a80a05). You can also configure the parameter with case-sensitive placeholder strings that are replaced with actual values if the 'Expand Value' parameter (see below) is set to **Yes**: ■ <MAC>: Replaced by the MAC address of the client. The MAC address is obtained from the client's DHCP request. For example, the parameter can be set to: http://192.168.3.155:5000/provisioning/cfg_<MAC>.txt ■ <IP>: Replaced by the IP address assigned by the DHCP server to the client. For example, the parameter can be set to: http://192.168.3.155:5000/provisioning/cfg_<IP>.txt
'Expand Value' `expand-value` [DhcpOption_ExpandValue]	Enables the use of the special placeholder strings, "<MAC>" and "<IP>" for configuring the 'Value' parameter (see above). ■ [0] **No** ■ [1] **Yes** (default) **Note:** The parameter is applicable only to values of type ASCII (see the 'Type' parameter, above).

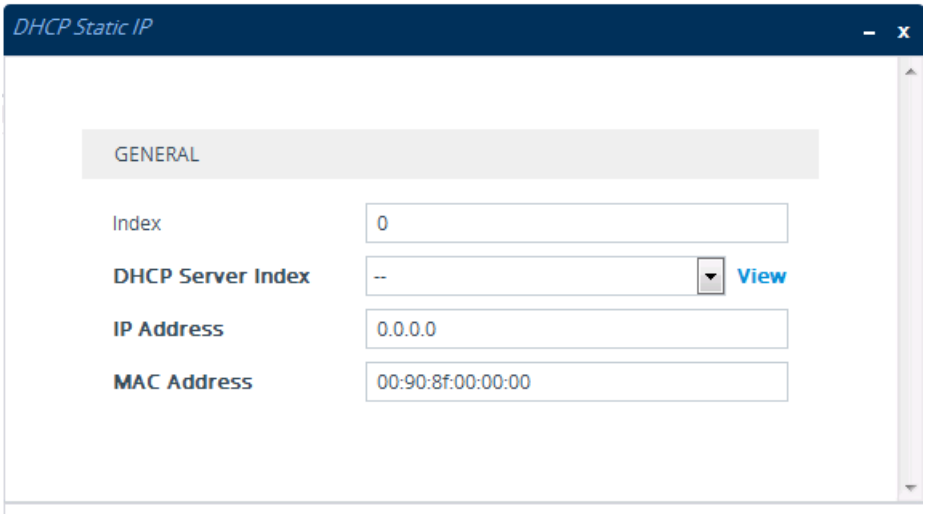
Configuring Static IP Addresses for DHCP Clients

The DHCP Static IP table lets you configure up to 100 DHCP clients with static IP addresses. The static IP address is a "reserved" IP address for a specified DHCP client defined by MAC address. In other words, instead of assigning the DHCP client with a different IP address upon each IP address lease renewal request, the DHCP server assigns the client the same IP address. For DHCP clients that are not listed in the table, the DHCP server assigns a random IP address from its address pool, as in normal operation.

The following procedure describes how to configure static IP addresses for DHCP clients through the Web interface. You can also configure it through ini file [DhcpStaticIP] or CLI (`configure network > dhcp-server static-ip <index>`).

➢ To configure static IP addresses for DHCP clients:

1. Open the DHCP Servers table (see Configuring the DHCP Server).
2. Select the row of the desired DHCP server for which you want to configure static IP addresses for DHCP clients, and then click the **DHCP Static IP** link located below the table; the DHCP Static IP table opens.
3. Click **New**; the following dialog box appears:



4. Configure a static IP address for a specific DHCP client according to the parameters described in the table below.
5. Click **Apply**.

Table 16-5: DHCP Static IP Table Parameter Descriptions

Parameter	Description
'Index' `dhcp static-ip <index>`	Defines an index number for the new table row. **Note:** Each row must be configured with a unique index.

Parameter	Description
[DhcpStaticIP_Index]	
'DHCP Server Index' `dhcp-server-number` [DhcpStaticIP_DhcpServerIndex]	Associates the DHCP Static IP table entry with a DHCP server that you configured in Configuring the DHCP Server. **Note:** Currently, only one DHCP server (Index 0) can be configured and therefore, the parameter is always set at 0.
'IP Address' `ip-address` [DhcpStaticIP_IPAddress]	Defines the "reserved", static IP address (IPv4) to assign the DHCP client. The default is 0.0.0.0.
'MAC Address' `mac-address` [DhcpStaticIP_MACAddress]	Defines the DHCP client by MAC address (in hexadecimal format). The valid value is a string of up to 20 characters. The format includes six groups of two hexadecimal digits, each separated by a colon. The default MAC address is 00:90:8f:00:00:00.

Viewing and Deleting DHCP Clients

The DHCP Clients table lets you view currently serviced DHCP clients by the DHCP server. The table also lets you delete DHCP clients. If you delete a client, the DHCP server ends the lease of the IP address to the client and the IP address becomes available for allocation by the DHCP server to another client.

The following procedure describes how to view DHCP clients through the Web interface. You can also view this through CLI:

- To view DHCP clients:

```
# show network dhcp clients
```

- To view DHCP clients according to IP address:

```
# show network dhcp ip
```

- To view DHCP clients according to MAC address:

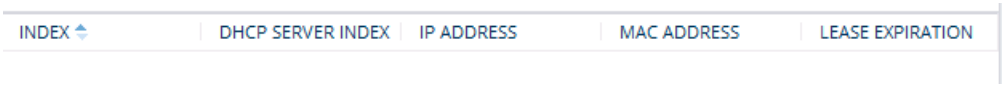
```
# show network dhcp mac
```

- To view DHCP clients that have been blacklisted from DHCP implementation (due to duplicated IP addresses in the network, where another device is using the same IP address as the one assigned to the client):

```
# show network dhcp black-list
```

➢ **To view or delete DHCP clients:**

1. Open the DHCP Servers table (see Configuring the DHCP Server).
2. Select the row of the desired DHCP server for which you want to view DHCP clients, and then click the **DHCP Clients** link located below the table; the DHCP Clients table opens:



The table displays the following per client:

- **Index:** Table index number.
- **DHCP Server Index:** The index number of the configured DHCP server scope in the DHCP Server table (see Configuring the DHCP Server) with which the client is associated.
- **IP Address:** IP address assigned to the DHCP client by the DHCP server.
- **MAC Address:** MAC address of the DHCP client.
- **Lease Expiration:** Date on which the lease of the DHCP client's IP address obtained from the DHCP server expires.

3. To delete a client:
   a. Select the table row index of the DHCP client that you want to delete.
   b. Click the **Action** button, and then from the drop-down menu, choose **Delete**; a confirmation message appears.
   c. Click **OK** to confirm deletion.

SIP-based Media Recording

The device can record SIP-based media (RTP/SRTP) call sessions traversing it. The device can record not only audio streams, but also video streams for audio-video calls. The media recording support is in accordance with the Session Recording Protocol (SIPRec), which describes architectures for deploying session recording solutions and specifies requirements for extensions to SIP that will manage delivery of RTP media to a recording device. The device's SIPRec feature is in compliance with the following:

- RFC 6341 (Use Cases and Requirements for SIP-Based Media Recording)
- Session Recording Protocol (draft-ietf-siprec-protocol-02)

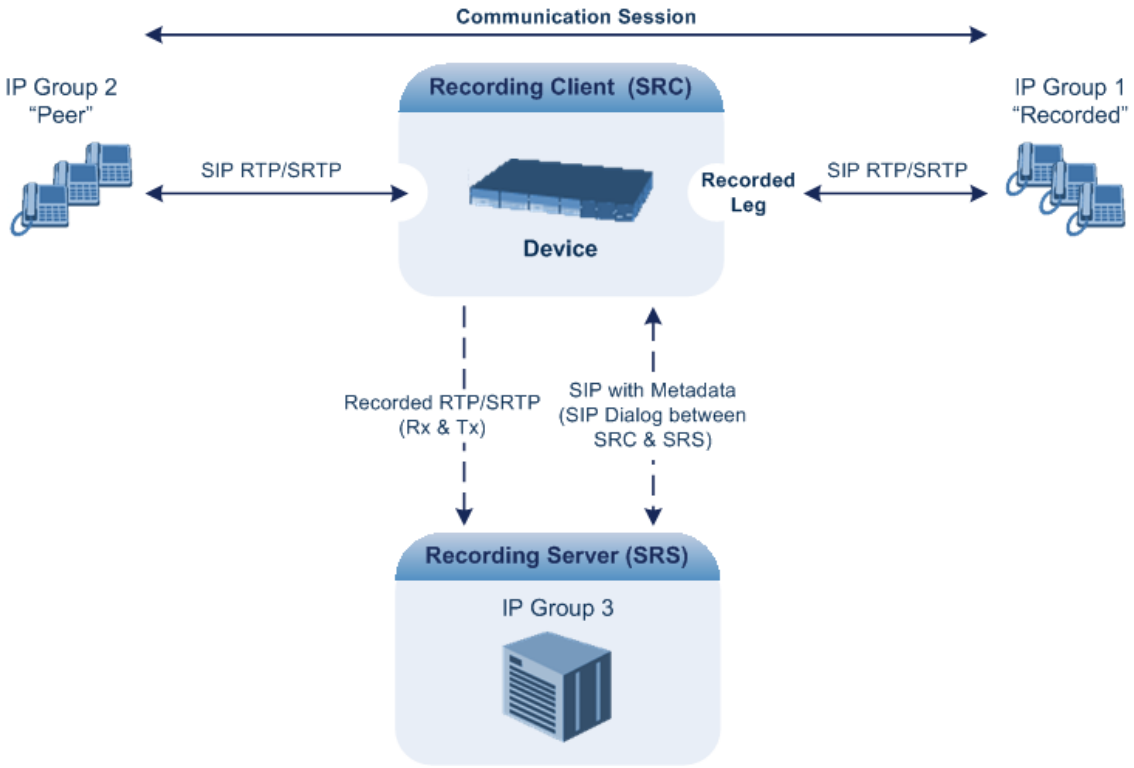
- Architecture (draft-ietf-siprec-architecture-03)
- RFC 7865 (Session Initiation Protocol (SIP) Recording Metadata)

Warning for Deployments in France: The device supports SIP-based Media Recording (SIPRec) according to RFC 6341. As such, you must adhere to the Commission Nationale Informatique et Liberté's (CNIL) directive (https://www.cnil.fr/en/rights-and-obligations) and be aware that article R226-15 applies penalties to the malicious interception, diversion, use or disclosure of correspondence sent, transmitted or received by means of telecommunication, or the setting up of a device designed to produce such interceptions.

- The SIP-based Media Recording feature is available only if the device is installed with a License Key (see License Key) that includes this feature. The License Key specifies the maximum number of supported SIP recording sessions. For audio-video calls, video recording needs additional SBC media channel resources.
- For maximum concurrent SIPRec sessions, refer to the device's *Release Notes*, which can be downloaded from AudioCodes website.
- You can view active and historical SIPRec call information, using the CLI command `show voip calls`.
- You can customize SBC CDRs generated by the device to include the field "Is Recorded" which indicates if the SBC leg was recorded or not. For more information, see Customizing CDRs for SBC Calls and Test Calls on page 1036.

Session recording is a critical requirement in many business communications environments such as call centers and financial trading floors. In some of these environments, all calls must be recorded for regulatory and compliance reasons. In others, calls may be recorded for quality control or business analytics. Recording is typically performed by sending a copy of the session media to the recording devices.

The SIPRec protocol specifies the use of SIP, SDP, and RTP to establish a Recording Session (RS) from the Session Recording Client (SRC), which is on the path of the Communication Session (CS), to a Session Recording Server (SRS) at the recording equipment. The device functions as the SRC, sending recording sessions to a third-party SRS, as shown in the figure below.

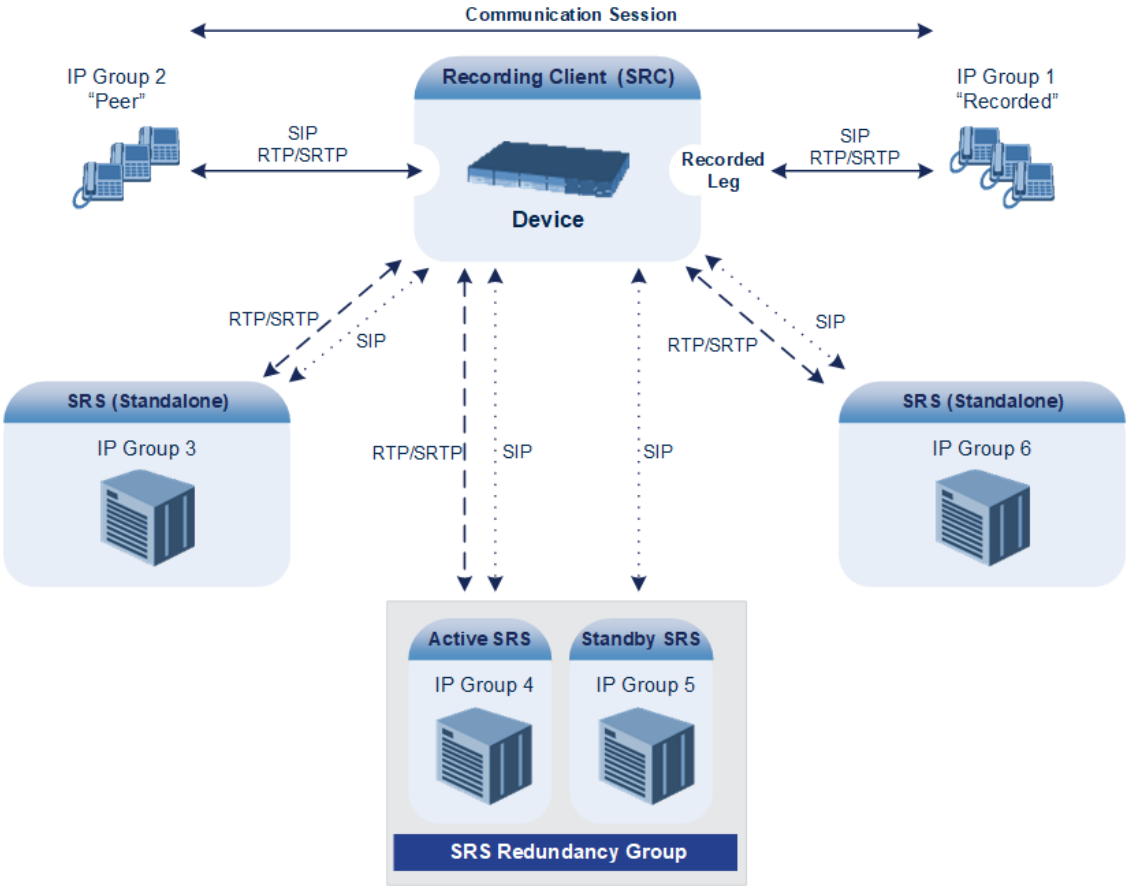


The device can record calls between two IP Groups. The type of calls to record can be specified by source and/or destination prefix number or SIP Request-URI, as well as by call initiator. The side ("leg") on which the recording is done must be specified. Specifying the leg is important as it determines the various call media attributes of the recorded RTP (or SRTP) such as coder type.

The device can also record SRTP calls and send it to the SRS in RTP, or vice versa. For this functionality, simply configure the 'SBC Media Security Mode' parameter of the IP Profile that is associated with the SRS's IP Group to **Secured** or **Not Secured**, respectively. If you need to record the call in a coder that is different to the coder used in the call, the device can also be located between an SRS and an SRC to perform coder transcoding. In this setup, the device receives SIP recording sessions from the SRC and transcodes the media between the SRC and SRS, and then forwards the recording to the SRS in the transcoded media format.

The device can send recorded SBC calls to multiple SRSs. To achieve this, you can configure up to three groups of SRSs, where each group can contain one SRS (standalone), or two SRSs operating in an active-standby (1+1) mode for SRS redundancy. The device sends both SIP signaling and RTP to all standalone SRSs.

For SRS redundancy, the device sends SIP signaling to all SRSs (active and standby), but sends RTP only to the active SRSs. If during a recorded call session, the standby SRS detects that the active SRS has gone offline, the standby SRS sends a re-INVITE to the device and the device then sends the recorded RTP to the standby SRS instead (which now becomes the active SRS). For new calls, if the device receives no response or a reject response from the active SRS to its' sent INVITE message, the device sends the recorded call to the standby SRS.


Communication Session
IP Group 2
"Peer"
SIP
RTP/SRTP
Recording Client (SRC)
Device
Recorded
Leg
SIP
RTP/SRTP
IP Group 1
"Recorded"
RTP/SRTP
SIP
RTP/SRTP
SIP
SRS (Standalone)
IP Group 3
RTP/SRTP
SIP
SIP
SRS (Standalone)
IP Group 6
Active SRS
IP Group 4
Standby SRS
IP Group 5
SRS Redundancy Group

- For the device's SRS active-standby feature to function, it must be supported by the third-party SRS. For supported third-party SRS vendors, contact your AudioCodes sales representative.
- The device can send recordings (media) to up to three active SRSs. In other words, any one of the following configurations are supported:
  - ✓ Up to three standalone (active) SRSs.
  - ✓ Up to three active-standby SRS pairs (i.e., six SRSs, but recordings are sent only to the three active SRSs).
  - ✓ One standalone (active) SRS and two active-standby SRS pairs.
  - ✓ Two standalone (active) SRSs and one active-standby SRS pair.
- SRS active-standby redundancy is a license-dependent feature and is available only if it is included in the License Key installed on the device (see Viewing the License Key on page 871). Therefore, the SIPRec feature can require two licenses – the regular license ("SIPRec Sessions") for standalone (active) SRSs and a license for SRS active-standby redundancy ("SIPRec Redundancy"). If you are implementing only standalone SRSs, you only need the “SIPRec Sessions” license. If you are implementing SRS active-standby redundancy, you need both licenses.
- The “SIPRec Sessions” license defines the maximum number of sessions for active SRSs (standalone SRS and the active SRS in the active-standby redundancy pair). The "SIPRec Redundancy" license defines the maximum number of SIPRec sessions for the standby SRS in the active-standby redundancy pair. For example, if you want to support 10 SIPRec sessions per SRS, the required licenses for various scenarios are as follows:
  - ✓ One standalone SRS: "SIPRec Sessions" = 10
  - ✓ Two standalone SRSs: "SIPRec Sessions" = 20
  - ✓ One active-standby redundancy pair: "SIPRec Sessions" = 10; "SIPRec Redundancy" = 10
  - ✓ Two active-standby redundancy pairs: "SIPRec Sessions" = 20; "SIPRec Redundancy" = 20
  - ✓ One standalone SRS and two active-standby redundancy pairs: "SIPRec Sessions" = 30; "SIPRec Redundancy" = 20

The device initiates a recording session by sending an INVITE message to the SRS when the call to be recorded is connected. The SIP From header contains the identity of the SRC and the To header contains the identity of the SRS. The SIP message body of the INVITE contains the following:

- SDP body:
  - Two 'm=' lines that represent the two RTP/SRTP streams (Rx and Tx).
  - Two 'a=label:' lines that identify the streams.

  If the recorded leg includes a video stream, the SDP not only includes the two audio streams ('m=audio'), but also two video streams ('m=video') in send-only RTP mode ('a=sendonly') - one for Tx and one for Rx.
- XML body (also referred to as metadata), which provides information on the participants of the call session:

- <group id>: Logging Session ID (displayed as [SID:nnnnn] in Syslog), converted to hex (or Base64 format). This number remains the same even if the call is forwarded or transferred. This is important for recorded calls.
- <session id>: SIP Call-ID header value of the recorded leg, which the device represents as a unique hashed number.
- <group-ref>: Same as <group id>.
- <participant id>: SIP From / To user.
- <nameID aor>: From/To user@host.
- <send> and <recv>: IDs for the RTP/SRTP streams in hex (or Base64 format) - bits 0-31 are the same as group, bits 32-47 are the RTP port.
- <stream id>: Same as <send> for each participant.
- <label>: 1 and 2 (same as in the SDP's 'a=label:' line).
- RFC 7865 only:
  - <sessionrecordingassoc>: Session association data.
  - <participantsessionassoc>: Data for association between participant and session.
  - <participantstreamassoc>: Data for association between participant and stream.

If the recorded leg includes a video stream, the metadata body contains two additional <stream> sections, which denote the Tx and Rx recording streams of the video payload. When RFC 7865 is chosen as the metadata format, the <participantstreamassoc> sections also contain this additional pair of streams.

You can configure the format of the recording metadata (i.e., based on RFC 7865 or "legacy") generated by the device. For more information, see Configuring Format of SIPRec Metadata on page 260.

The SRS can respond with 'a=recvonly' for immediate recording or 'a=inactive' if recording is not yet needed, and send a re-INVITE at any later stage with the desired RTP/SRTP mode change. If a re-INVITE is received in the original call (e.g., when a call is on hold), the device sends another re-INVITE with two 'm=' lines to the SRS with the updated RTP/SRTP data. If the recorded leg uses SRTP, the device can send the media streams to the SRS as SRTP; otherwise, the media streams are sent as RTP to the SRS.

Below is an example of an INVITE sent by the device to the SRS, showing the legacy and RFC 7865 metadata formats (only one of these is generated in real-life scenarios):

```
INVITE sip:VSRP@1.9.64.253 SIP/2.0
Via: SIP/2.0/UDP 192.168.241.44:5060;branch=z9hG4bKac505782914
Max-Forwards: 10
From: <sip:192.168.241.44>;tag=1c505764207
To: <sip:VSRP@1.9.64.253>
Call-ID: 505763097241201011157@192.168.241.44
CSeq: 1 INVITE
```

```
Contact: <sip:192.168.241.44:5060>;src
Supported: replaces,resource-priority
Allow:
REGISTER,OPTIONS,INVITE,ACK,CANCEL,BYE,NOTIFY,PRACK,REFER,INF
O,SUBSCRIBE,UPDATE
Require: siprec
User-Agent: Device /7.20A.258.980
Content-Type: multipart/mixed;boundary=boundary_ac1fffff85b
Content-Length: 1832
--boundary_ac1fffff85b
Content-Type: application/sdp

v=0
o=AudioCodesGW 921244928 921244893 IN IP4 10.33.8.70
s=SBC-Call
c=IN IP4 10.33.8.70
t=0 0
m=audio 6020 RTP/AVP 8 96
c=IN IP4 10.33.8.70
a=ptime:20
a=sendonly
a=label:1
a=rtpmap:8 PCMA/8000
a=rtpmap:96 telephone-event/8000
a=fmtp:96 0-15
m=audio 6030 RTP/AVP 8 96
c=IN IP4 10.33.8.70
a=ptime:20
a=sendonly
a=label:2
a=rtpmap:8 PCMA/8000
a=rtpmap:96 telephone-event/8000
a=fmtp:96 0-15
--boundary_ac1fffff85b

Content-Type: application/rs-metadata
Content-Disposition: recording-session
```

- **Legacy XML metadata:**

```
<?xml version="1.0" encoding="UTF-8"?>
<recording xmlns='urn:ietf:params:xml:ns:recording'>
 <datamode>complete</datamode>
 <group id="00000000-0000-0000-0000-00003a36c4e3">
  <associate-time>2010-01-24T01:11:57Z</associate-time>
```

```
 </group>
 <session id="0000-0000-0000-0000-00000000d0d71a52">
  <group-ref>00000000-0000-0000-0000-00003a36c4e3</group-ref>
  <start-time>2010-01-24T01:11:57Z</start-time>
    <ac:AvayaUCID
xmlns="urn:ietf:params:xml:ns:Avaya">FA080030C4E34B5B9E59</ac:Avaya
UCID>
 </session>
 <participant id="1056" session="0000-0000-0000-0000-
00000000d0d71a52">
  <nameID aor="1056@192.168.241.20"></nameID>
  <associate-time>2010-01-24T01:11:57Z</associate-time>
  <send>00000000-0000-0000-0000-1CF23A36C4E3</send>
  <recv>00000000-0000-0000-0000-BF583A36C4E3</recv>
 </participant>
    <participant id="182052092" session="0000-0000-0000-0000-
00000000d0d71a52">
  <nameID aor="182052092@voicelab.local"></nameID>
  <associate-time>2010-01-24T01:11:57Z</associate-time>
  <recv>00000000-0000-0000-0000-1CF23A36C4E3</recv>
  <send>00000000-0000-0000-0000-BF583A36C4E3</send>
 </participant>
 <stream id="00000000-0000-0000-0000-1CF23A36C4E3" session="0000-
0000-0000-0000-00000000d0d71a52">
  <label>1</label>
 </stream>
 <stream id="00000000-0000-0000-0000-BF583A36C4E3" session="0000-
0000-0000-0000-00000000d0d71a52">
  <label>2</label>
 </stream>
</recording>
--boundary_ac1fffff85b--
```

- **RFC 7865 XML metadata:**

```
<?xml version="1.0" encoding="UTF-8"?>
<recording xmlns="urn:ietf:params:xml:ns:recording" xmlns:ac="http://abc">
 <datamode>complete</datamode>
 <group group_id="4gAAAAC9YRUBDQDw">
  <associate-time>2018-04-17T09:35:41</associate-time>
 </group>
 <session session_id="OWc4Md2PHao=">
  <group-ref>4gAAAAC9YRUBDQDw</group-ref>
 </session>
```

```
 <participant participant_id="MjAw">
  <nameID aor="200@10.33.8.52">
     <name xml:lang="en">Bob</name>
  </nameID>
 </participant>
 <participant participant_id="MTAw">
  <nameID aor="100@10.33.8.52"></nameID>
 </participant>
 <stream stream_id="mBfiAAAAAL1hFQENAPA=" session_
id="OWc4Md2PHao=">
  <label>1</label>
 </stream>
 <stream stream_id="hBfiAAAAAL1hFQENAPA=" session_
id="OWc4Md2PHao=">
  <label>2</label>
 </stream>
 <sessionrecordingassoc session_id="OWc4Md2PHao=">
  <associate-time>2018-04-17T09:35:41</associate-time>
 </sessionrecordingassoc>
 <participantsessionassoc participant_id="MjAw" session_
id="OWc4Md2PHao=">
  <associate-time>2018-04-17T09:35:41</associate-time>
 </participantsessionassoc>
 <participantsessionassoc participant_id="MTAw" session_
id="OWc4Md2PHao=">
  <associate-time>2018-04-17T09:35:41</associate-time>
 </participantsessionassoc>
 <participantstreamassoc participant_id="MjAw">
  <send>mBfiAAAAAL1hFQENAPA=</send>
  <recv>hBfiAAAAAL1hFQENAPA=</recv>
 </participantstreamassoc>
 <participantstreamassoc participant_id="MTAw">
  <send>hBfiAAAAAL1hFQENAPA=</send>
  <recv>mBfiAAAAAL1hFQENAPA=</recv>
 </participantstreamassoc>
</recording>
```

Configuring SIP Recording Rules

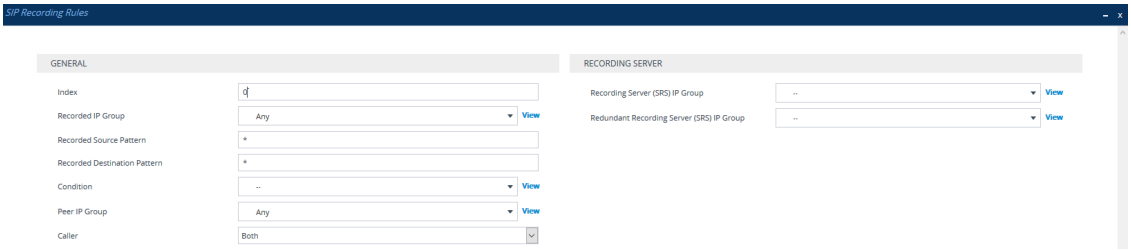
The SIP Recording Rules table lets you configure up to 30 SIP-based media recording rules. A SIP Recording rule defines call routes that you want to record. For an overview of the feature, see SIP-based Media Recording.

- To view the total number of currently active SIPRec signaling sessions, use the CLI command **show voip calls statistics siprec**. For more information, refer to the *CLI Reference Guide*.
- To configure the device's timestamp format (local or UTC) in SIP messages sent to the SRS, see the SIPRecTimeStamp parameter.
- When recording SRTP-to-SRTP calls, if you want to send the recorded media to the SRS as RTP (i.e., decrypted), you need to add an IP Profile for the SRS and configure its 'SBC Media Security Mode' parameter to **Not Secured** (see Configuring IP Profiles on page 519).
- If you configure a SIP Recording rule (see SIP-based Media Recording on page 247) for calls that have also been configured for direct media (media bypass), using a SIP Interface ('Direct Media' parameter) or an IP Profile ('Direct Media Tag' parameter), the device automatically disables direct media for these calls (during their SIP signaling setup). This ensures that the media passes through the device so that it can be recorded and sent to the SRS. However, if you enable direct media using the [SBCDirectMedia] global parameter (i.e., for all calls), direct media is always enforced and calls will **not** be recorded.

The following procedure describes how to configure SIP Recording rules through the Web interface. You can also configure it through ini file [SIPRecRouting] or CLI (`configure voip > sip-definition sip-recording sip-rec-routing`).

➢ To configure a SIP Recording rule:

1. Open the SIP Recording Rules table (**Setup** menu > **Signaling & Media** tab > **SIP Recording** folder > **SIP Recording Rules**).
2. Click **New**; the following dialog box appears:



The following configuration records calls made by IP Group "ITSP" to IP Group "IP-PBX" that have the destination number prefix "1800". The device records the calls from the leg interfacing with IP Group "IP PBX" (peer) and sends the recorded media to IP Group "SRS-1". SRS redundancy has also been configured, where IP Group "SRS-1" is the active SRS and IP Group "SRS-2" the standby SRS.

- Recorded IP Group: "ITSP"
- Recorded Destination Pattern: 1800
- Peer IP Group: "IP-PBX"
- Caller: Peer Party
- Recording Server (SRS) IP Group: "SRS-1"

- Recording Server (SRS) IP Group: "SRS-2"

1. Configure a SIP recording rule according to the parameters described in the table below.
2. Click **Apply**, and then save your settings to flash memory.

Table 16-6: SIP Recording Rules Table Parameter Descriptions

Parameter	Description
General	
'Index' [SIPRecRouting_Index]	Defines an index number for the new table record.
'Recorded IP Group' `recorded-ip-group-name` [SIPRecRouting_ RecordedIPGroupName]	Defines the IP Group participating in the call and the recording is done on the leg interfacing with this IP Group. To configure IP Groups, see Configuring IP Groups. By default, all IP Groups are defined (**Any**). **Note:** ■ The parameter is mandatory. ■ For an SBC RTP-SRTP session, the recorded IP Group must be set to the RTP leg if recording is required to be RTP, or set to the SRTP leg if recording is required to be SRTP.
'Recorded Source Pattern' `recorded-src-pattern` [SIPRecRouting_ RecordedSourcePrefix]	Defines calls to record based on source number or SIP URI. You can use special patterns (notations) to denote the number or URI. For example, if you want to match this rule to user parts whose last four digits (i.e., suffix) are 4 followed by any three digits (e.g., 4008), then configure this parameter to "(4xxx)". For available patterns, see Patterns for Denoting Phone Numbers and SIP URIs on page 1121. The default value is the asterisk (*) symbol, meaning any source number or URI.
'Recorded Destination Pattern' `recorded-dst-prefi` [SIPRecRouting_ RecordedDestinationPrefix]	Defines calls to record based on destination number or URI. You can use special patterns (notations) to denote the number or URI. For example, if you want to match this rule to user parts whose last four digits (i.e., suffix) are 4 followed by any three digits (e.g., 4008), then configure this parameter to "(4xxx)". For available

Parameter	Description
	patterns, see Patterns for Denoting Phone Numbers and SIP URIs on page 1121. The default value is the asterisk (*) symbol, meaning any destination number or URI.
'Condition' `condition-name` [SIPRecRouting_ ConditionName]	Assigns a Message Condition rule to the SIP Recording rule to base the start of call recording on a specific condition. To configure Message Condition rules, see Configuring Message Condition Rules on page 642. For more information on using conditions with SIPRec, see Using Conditions for Starting a SIPRec Session on the next page.
'Peer IP Group' `peer-ip-group-name` [SIPRecRouting_ PeerIPGroupName]	Defines the peer IP Group that is participating in the call. By default, all IP Groups are defined (**Any**).
'Caller' `caller` [SIPRecRouting_Caller]	Defines which calls to record according to which party is the caller. ■ [0] **Both** = (Default) Caller can be peer or recorded side ■ [1] **Recorded Party** ■ [2] **Peer Party**
Recording Server	
'Recording Server (SRS) IP Group' `srs-ip-group-name` [SIPRecRouting_ SRSIPGroupName]	Defines the IP Group of the recording server (SRS). By default, no value is defined. **Note:** ■ The parameter is mandatory.
'Redundant Recording Server (SRS) IP Group' `srs-red-ip-group-name` [SIPRecRouting_ SRSRedundantIPGroupName]	Defines the IP Group of the redundant SRS in the active-standby pair for SRS redundancy. By default, no value is defined. **Note:** ■ The IP Group of the redundant SRS must be different to the IP Group of the main SRS (see 'Recording Server (SRS) IP Group' parameter).

Using Conditions for Starting a SIPRec Session

You can start and stop the recording of calls (SIPRec) based on user-defined conditions. The condition is configured as a Message Condition rule in the Message Conditions table, which is then assigned to the SIP Recording rule in the SIP Recording Rules table. Only if the condition is met will the device start recording the call. The feature is typically configured using Message Condition rules together with Call Setup rules.

For this feature, you can use only the following keywords for the syntax of the Message Condition rule:

- var.global
- var.session.0
- srctags/dsttags

For more information on using the above syntax for message manipulation, refer to the *Syntax for SIP Message Manipulation Reference Guide*.

The following procedure provides a SIP Recording configuration example for using a condition with the "srctags" keyword to start recording a call for IP Group "ITSP" if the incoming SIP message contains the header, "X-Record:yes".

➢ **To use conditions for SIPRec:**

1. In the Call Setup Rules table (see Configuring Call Setup Rules on page 595), click **New**, and then configure a Call Setup rule with the following properties:
   - 'Index': 0
   - 'Rules Set ID': 1
   - 'Condition': header.X-Record==’yes’
   - 'Action Subject': srctags
   - 'Action Type': Modify
   - 'Action Value': 'record'
1. In the IP Groups table (see Configuring IP Groups on page 451), assign the Call Setup rule that you configured in the previous step to the IP Group that you want to record (i.e., the "Recorded IP Group"):
   - 'Call Setup Rules Set ID': 1
2. In the Message Conditions table (see Configuring Message Condition Rules on page 642), click **New**, and then configure a Message Condition rule with the following properties:
   - 'Index': 0
   - 'Name': CallRec
   - 'Condition': srctags == 'record'

3. In the SIP Recording Rules table, configure a SIP Recording rule as desired and assign it the Message Condition rule that you configured in the previous step:
   - 'Recorded IP Group': ITSP
   - 'Condition': CallRec

Configuring Format of SIPRec Metadata

You can configure the format of the XML-based recording metadata that the device generates and includes in the SIP messages that it sends to the recording server (SRS). It is important that the device generates the metadata in a format that is acceptable by the SRS.

The device supports the following formats:

- **RFC 7865** - the device generates the recording metadata in a format that is according to RFC 7865, whereby all IDs (e.g., participant ID) are in Base64 format. This metadata format also includes additional XML tags with association information (e.g., "<participantsessionassoc>").
- **Legacy** (default) - The device generates the recording metadata in a "legacy" format, whereby the user part of the participant URI (source or destination) is used as the ID.

➢ **To configure the format of the metadata:**

1. Open the SIP Recording Settings page (**Setup** menu > **Signaling & Media** tab > **SIP Recording** folder > **SIP Recording Settings**).

Figure 16-1: Configuring SIPRec Metadata Format

2. From the 'SIP Recording Metadata Format' drop-down list, select the desired format.
3. Click **Apply**.

Configuring Video Recording Synchronization

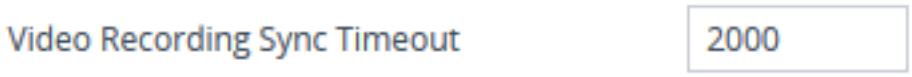
If you also want to record the video stream of audio-video calls, you need to configure a video synchronization timeout. When the video stream is also recorded, the device operates as follows:

1. Once the call is answered by the called UA (i.e., connected), the UAs' audio streams are connected and the device sends a SIP INVITE to the SRS. However, for correct video synchronization, the UAs' video streams are not yet connected at this stage.
2. When a SIP 200 OK response is received from the SRS and the UAs' ports have been negotiated, the device connects all video streams - the UAs' video stream and the recorded video stream (the recorded audio stream is also sent to the SRS at this stage). However, if the 200 OK from the SRS is not received within a user-defined video synchronization timeout, the device connects the video stream between the UAs.

➢ **To configure video synchronization timeout:**

1. Open the SIP Recording Settings page (**Setup** menu > **Signaling & Media** tab > **SIP Recording** folder > **SIP Recording Settings**).

Figure 16-2: Configuring Video Recording Synchronization



2. In the 'Video Recording Sync Timeout' field, enter a timeout for receiving the SIP 200 OK from the SRS.
3. Click **Apply**.

On-Demand SIPRec Sessions

The device supports on-demand SIPRec sessions. It can be triggered to start or stop recording at any stage of a connected call. In other words, it can perform multiple recordings per call.

Starting and stopping recording is triggered by the receipt of a SIP INFO message containing AudioCodes proprietary X-AC-Action header with the 'start-siprec' parameter for starting recording, or 'stop-siprec' for stopping recording. The following shows an example of an incoming SIP INFO message that triggers a SIPRec session:

```
INFO sip:alice@pc33.example.com SIP/2.0
Via: SIP/2.0/UDP 192.0.2.2:5060;branch=z9hG4bKnabcdef
To: Bob <sip:bob@example.com>;tag=a6c85cf
From: Alice <sip:alice@example.com>;tag=1928301774
Call-Id: a84b4c76e66710@pc33.example.com
CSeq: 314333 INFO
X-AC-Action: start-siprec
Content-Length: 0
```

The only required configuration is to add a Message Manipulation rule that adds the IP Group of the SRS to the X-AC-Action header in the incoming SIP INFO message, using the 'recording-ip-group' parameter:

```
X-AC-Action: start-siprec;recording-ip-group="<SRS IP Group>"
```

- or -

```
X-AC-Action: start-siprec;recording-ip-group="<SRS IP Group>";recorded-side=peer
```

Where:

- 'recording-ip-group' indicates the name of the recording IP Group (i.e., SRS). The value of this parameter can be enclosed in quotes or without quotes.
- 'recorded-side=peer' is an optional parameter that indicates recording should be done on the peer side (i.e., not the side receiving the X-AC-Action header).

Configure the Message Manipulation rule in the Message Manipulations table (see Configuring SIP Message Manipulation on page 634) as follows:

- 'Name': IP Group of SRS for Start SIPREC
- 'Manipulation Set ID': 0
- 'Message Type': **Info**
- 'Condition': **Header.X-AC-Action contains 'start-siprec'**
- 'Action Subject': **Header.X-AC-Action**
- 'Action Type': **Modify**
- 'Action Value': **'start-siprec;recording-ip-group=\"IPGroup_2\"'**

Assign the above Message Manipulation Set ID to the IP Group of the sender of the SIP INFO message (i.e., not the SRS IP Group).

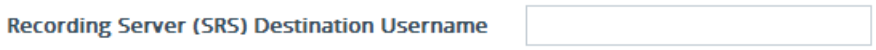
- You don't need to configure anything in the SIP Recording Rules table for this feature.
- The device rejects the INFO message (with a 500 response) if any of the following scenarios exist (call is not affected):
  - ✓ The 'recording-ip-group' parameter doesn't exist.
  - ✓ The X-AC-Action header contains no parameters.
  - ✓ Starting and stopping recording were not **both** triggered by a SIP INFO message, as described in this section. In other words, if starting recording was triggered by the normal SIP Recording Rules table and stopping recording was triggered by a SIP INFO message containing the X-AC-Action header with the 'stop-siprec' parameter, the device rejects the INFO message.
  - ✓ The X-AC-Action header contains the 'stop-siprec' parameter even though SIP recording was not started.
  - ✓ The X-AC-Action header contains the 'start-siprec' parameter on a recorded-side while there is an active SIPRec session on the other side.
  - ✓ Three recording sessions are already in progress (maximum).
- The device forwards information in SIP INFO messages such as the transcript of the call from the SRS to the initiator of the on-demand SIPRec session.

Configuring SIP User Part for SRS

You can configure the SIP user part of the Request-URI for the recording server (SRS). The device inserts this user part in the SIP To header of the INVITE message sent to the SRS.

➢ **To configure the SIP user part for SRS:**

1. Open the SIP Recording Settings page (**Setup** menu > **Signaling & Media** tab > **SIP Recording** folder > **SIP Recording Settings**).



2. In the 'Recording Server (SRS) Destination Username' field, enter a user part value (string of up to 50 characters).
3. Click **Apply**.

Interworking SIP-based Media Recording with Third-Party Vendors

The device can interwork the SIP-based Media Recording feature with third-party vendors, as described in the following subsections.

SIPRec with Genesys Equipment

The device's SIP-based media recording can interwork with Genesys' equipment. Genesys sends its proprietary X-Genesys-CallUUID header (which identifies the session) in the first SIP message, typically in the INVITE and the first 18x response. If the device receives a SIP message with Genesys SIP header, it adds the header's information to theAudioCodes' proprietary tag in the XML metadata of the SIP INVITE that it sends to the recording server, as shown below:

```
<ac:GenesysUUID
xmlns="urn:ietf:params:xml:ns:Genesys">4BOKLLA3VH66JF112M1CC9VHKS14
F0KP</ac:GenesysUUID>
```

No configuration is required for this support.

SIPRec with Avaya Equipment

The device's SIP-based media recording can interwork with Avaya equipment. The Universal Call Identifier (UCID) is Avaya's proprietary call identifier used to correlate call records between different systems and identifies sessions. Avaya generates this in outgoing calls. If the device receives a SIP INVITE from Avaya, it adds the UCID value, received in the User-to-User SIP header to the AudioCodes proprietary tag in the XML metadata of the SIP INVITE that it sends to the recording server. For example, if the received SIP header is:

```
User-to-User: 00FA080019001038F725B3;encoding=hex
```

the device includes the following in the XML metadata:

```
xml metadata:
<ac:AvayaUCID xmlns="urn:ietf:params:xml:ns:Avaya">
FA080019001038F725B3</ac:AvayaUCID>
```

For calls sent from the device to Avaya equipment, the device can generate the Avaya UCID, if required. To configure this support, use the following parameters:

- 'UUI Format' in the IP Groups table - enables Avaya support.
- 'Network Node ID' - defines the Network Node Identifier of the device for Avaya UCID.

Customizing Recorded SIP Messages Sent to SRS

The original SIP headers of recorded legs are not included in the INVITE messages that the device sends to the SRS. If you need to include SIP headers, you can use Message Manipulation rules (see Configuring SIP Message Manipulation on page 634) to add them to these INVITE messages. The following examples describe how to configure this using Message Manipulation rules:

- **Example 1** - Adding a specific SIP header called "My-header" to the INVITE that is sent to the SRS:

  a. The example uses two Message Manipulation rules - one for storing the header by using manipulation syntax for session variables, and one for adding the header to the INVITE.

Parameter	Value
Index	0
Name	Store My-header in var.session
Manipulation Set ID	11
Message Type	Any
Condition	Header.My-header exists And Header.My-header != ''
Action Subject	Var.Session.0
Action Type	Modify
Action Value	Header.My-header
Index	1
Name	Send My-header to SRS

Parameter	Value
Manipulation Set ID	12
Message Type	Invite.Request
Condition	Var.Session.0 != ''
Action Subject	Header.My-header
Action Type	Add
Action Value	Var.Session.0

b. Assign the above manipulation rules to the relevant IP Groups:

- In the IP Group of the recorded call leg which sends this header, configure the 'Inbound Message Manipulation Set' parameter to 11 (i.e., rule configured in Index 0).
- In the IP Group of the SRS, configure the 'Outbound Message Manipulation Set' parameter to 12 (i.e., rule configured in Index 1).

- **Example 2** - Adding multiple (three) SIP headers called "My-header1", "My-header2" and "My-header3" to the INVITE that is sent to the SRS:

a. The example uses regex (regular expression) with manipulation rules for extracting each header (a comma is used to separate headers).

Parameter	Value
Index	0
Name	Store headers in var.session
Manipulation Set ID	11
Message Type	Any
Condition	Header.My-header1 exists And Header.My-header2 exists And Header.My-header3 exists
Action Subject	Var.Session.0
Action Type	Modify
Action Value	Header.My-header1+','+ Header.My-header2+','+ Header.My-header3

Parameter	Value
Row Rule	Use Current Condition
Index	1
Name	Send My-header1 to SRS
Manipulation Set ID	12
Message Type	Invite.Request
Condition	Var.Session.0 regex (.*),(.*),(.*)
Action Subject	Header.My-header1
Action Type	Add
Action Value	$1
Row Rule	Use Current Condition
Index	2
Name	Send My-header2 to SRS
Manipulation Set ID	12
Message Type	Invite.Request
Condition	Var.Session.0 regex (.*),(.*),(.*)
Action Subject	Header.My-header2
Action Type	Add
Action Value	$2
Row Rule	Use Previous Condition
Index	3
Name	Send My-header3 to SRS
Manipulation Set ID	12
Message Type	Invite.Request
Condition	Var.Session.0 regex (.*),(.*),(.*)

Parameter	Value
Action Subject	Header.My-header3
Action Type	Add
Action Value	$3
Row Rule	Use Previous Condition

b. Assign the above manipulation rules to the relevant IP Groups:

- In the IP Group of the recorded call leg which sends this header, configure the 'Inbound Message Manipulation Set' parameter to 11 (i.e., rule configured in Index 0).
- In the IP Group of the SRS, configure the 'Outbound Message Manipulation Set' parameter to 12 (i.e., rules configured in Index 1, 2 and 3).

RADIUS-based Services

The device supports Remote Authentication Dial In User Service (RADIUS) by acting as a RADIUS client. You can use RADIUS for the following:

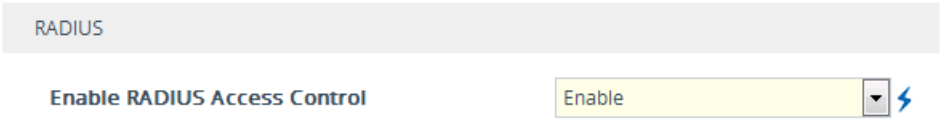
- Authentication and authorization of management users (login username and password) to gain access to the device's management interface.
- Accounting where the device sends accounting data of SIP calls as call detail records (CDR) to a RADIUS Accounting server (for third-party billing purposes).

Enabling RADIUS Services

Before you can implement any RADIUS services, you must enable the RADIUS feature, as described in the procedure below.

➢ **To enable RADIUS:**

1. Open the Authentication Server page (**Setup** menu > **Administration** tab > **Web & CLI** folder > **Authentication Server**).



2. Under the RADIUS group, from the 'Enable RADIUS Access Control' drop-down list, select **Enable.**
3. Click **Apply**, and then reset the device with a save-to-flash for your settings to take effect.

Configuring RADIUS Servers

The RADIUS Servers table lets you configure up to three RADIUS servers. You can use RADIUS servers for RADIUS-based management-user login authentication and/or RADIUS-based accounting (sending of SIP CDRs to the RADIUS server).

When multiple RADIUS servers are configured, RADIUS server redundancy can be implemented. When the primary RADIUS server is down, the device sends a RADIUS request twice (one retransmission) and if both fail (i.e., no response), the device considers the server as down and attempts to send requests to the next server. The device continues sending RADIUS requests to the redundant RADIUS server even if the primary server returns to service later on. However, if a device reset occurs or an HA switchover occurs in a High-Availability (HA) system, the device sends RADIUS requests to the primary RADIUS server. By default, the device waits for up to two seconds (i.e., timeout) for a response from the RADIUS server for RADIUS requests and retransmission before it considers the server as down.

For each RADIUS server, the IP address, port, and shared secret can be configured. Each RADIUS server can be defined for RADIUS-based login authentication and/or RADIUS-based accounting. By setting the relevant port (authentication or accounting) to "0" disables the corresponding functionality. If both ports are configured, the RADIUS server is used for authentication and accounting. All servers configured with non-zero Authorization ports form an Authorization redundancy group and the device sends authorization requests to one of them, depending on their availability. All servers configured with non-zero Accounting ports form an Accounting redundancy group and the device sends accounting CDRs to one of them, depending on their availability. Below are example configurations:

- Only one RADIUS server is configured and used for authorization and accounting purposes (no redundancy). Therefore, both the Authorization and Accounting ports are defined.
- Three RADIUS servers are configured:
  - Two servers are used for authorization purposes only, providing redundancy. Therefore, only the Authorization ports are defined, while the Accounting ports are set to 0.
  - One server is used for accounting purposes only (i.e., no redundancy). Therefore, only the Accounting port is defined, while the Authorization port is set to 0.
- Two RADIUS servers are configured and used for authorization and accounting purposes, providing redundancy. Therefore, both the Authorization and Accounting ports are defined.

The status of the RADIUS severs can be viewed through CLI:

```
# show system radius servers status
```

The example below shows the status of two RADIUS servers in redundancy mode for authorization and accounting:

```
servers 0
 ip-address 10.4.4.203
 auth-port 1812
 auth-ha-state "ACTIVE"
 acc-port 1813
 acc-ha-state "ACTIVE"
servers 1
 ip-address 10.4.4.202
 auth-port 1812
 auth-ha-state "STANDBY"
 acc-port 1813
 acc-ha-state "STANDBY"
```

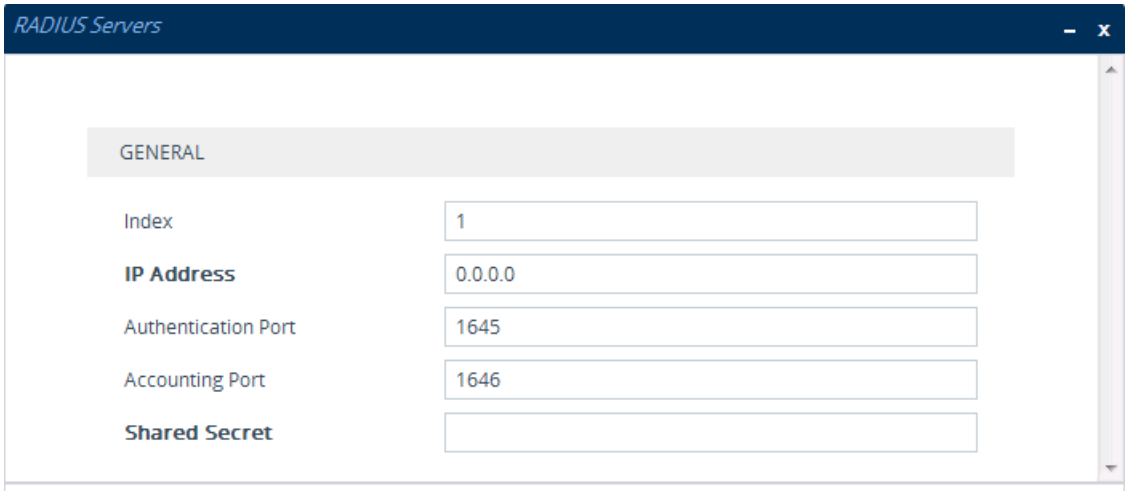
Where *auth-ha-state* and *acc-ha-state* display the authentication and accounting redundancy status respectively. "ACTIVE" means that the server was used for the last sent authentication or accounting request; "STANDBY" means that the server was not used in the last sent request.

- To enable and configure RADIUS-based accounting, see Configuring RADIUS Accounting.
- The device can send up to 201 concurrent RADIUS requests per RADIUS service type (Accounting or Authentication), per RADIUS server (up to three servers per service type), and per local port (up to 2 local ports). For example, 402 (201 * 2) concurrent RADIUS requests can be sent for Authentication and 402 (201 * 2) for Accounting.

The following procedure describes how to configure a RADIUS server through the Web interface. You can also configure it through ini file [RadiusServers] or CLI (`configure system > radius servers`).

➢ **To configure a RADIUS server:**

1. Open the RADIUS Servers table (**Setup** menu > **IP Network** tab > **RADIUS & LDAP** folder > **RADIUS Servers**).
2. Click **New**; the following dialog box appears:



3. Configure a RADIUS server according to the parameters described in the table below.
4. Click **Apply**.

Table 16-7: RADIUS Servers Table Parameter Descriptions

Parameter	Description
'Index' [RadiusServers_Index]	Defines an index number for the new table row. **Note:** Each row must be configured with a unique index.
'IP Address' `ip-address` [RadiusServers_ IPAddress]	Defines the IP address (IPv4) of the RADIUS server.
'Authentication Port' `auth-port` [RadiusServers_ AuthenticationPort]	Defines the port of the RADIUS Authentication server for authenticating the device with the RADIUS server. When set to any value other than 0, the RADIUS server is used by the device for RADIUS-based management-user login authentication. When set to 0, RADIUS-based login authentication is not implemented. The valid value is 0 to any integer. The default is 1645.
'Accounting Port' `acc-port` [RadiusServers_ AccountingPort]	Defines the port of the RADIUS Accounting server to where the device sends accounting data of SIP calls as call detail records (CDR). When set to any value other than 0, the RADIUS server is used by the device for RADIUS-based accounting (CDR). When set to 0, RADIUS-based accounting is not implemented. The valid value is 0 to any integer. The default is 1646.
'Shared Secret' `shared-secret` [RadiusServers_ SharedSecret]	Defines the shared secret (password) for authenticating the device with the RADIUS server. This should be a cryptically strong password. The shared secret is also used by the RADIUS server to verify the authentication of the RADIUS messages sent

Parameter	Description
	by the device (i.e., message integrity). The valid value is up to 48 characters. By default, no value is defined. **Note:** The password cannot be configured with wide characters.

Configuring Interface for RADIUS Communication

The device can communicate with the RADIUS server through its' OAMP (default) or SIP Control network interface. To change the interface for RADIUS traffic, use the [RadiusTrafficType] parameter.

- If you configure the parameter to Control, make sure that only one Control interface is configured in the IP Interfaces table (see Configuring IP Network Interfaces); otherwise, RADIUS communication fails.

Configuring RADIUS Packet Retransmission

You can configure the device to resend packets to the RADIUS server if no response is received from the server. This functionality is applicable to RADIUS-based user authentication and RADIUS-based accounting.

➢ **To configure RADIUS packet retransmission:**

1. Open the Authentication Server page (**Setup** menu > **Administration** tab > **Web & CLI** folder > **Authentication Server**).

2. Under the RADIUS group, do the following:
   - In the 'RADIUS Packets Retransmission' field (RADIUSRetransmission), enter the maximum number of RADIUS retransmissions that the device performs if no response is received from the RADIUS server.
   - In the 'RADIUS Response Time Out' field (RadiusTO), enter the interval (in seconds) that the device waits for a response before sending a RADIUS retransmission.
3. Click **Apply**.

Configuring the RADIUS Vendor ID

The vendor-specific attribute (VSA) identifies the device to the RADIUS server using the Vendor ID (as registered with the Internet Assigned Numbers Authority or IANA). The device's default vendor ID is 5003 which can be changed, as described in the following procedure. For an

example of using the Vendor ID, see Setting Up a Third-Party RADIUS Server. The procedure is applicable to both RADIUS-based user authentication and RADIUS-based accounting.

> The Vendor ID must be the same as the Vendor ID set on the third-party RADIUS server. See the example for setting up a third-party RADIUS server in Setting Up a Third-Party RADIUS Server.

➢ **To configure the RADIUS Vendor ID:**

1. Open the Authentication Server page (**Setup** menu > **Administration** tab > **Web & CLI** folder > **Authentication Server**).

2. Under the RADIUS group, in the 'RADIUS VSA Vendor ID' field, enter the **same** vendor ID number as set on the third-party RADIUS server.
3. Click **Apply**.

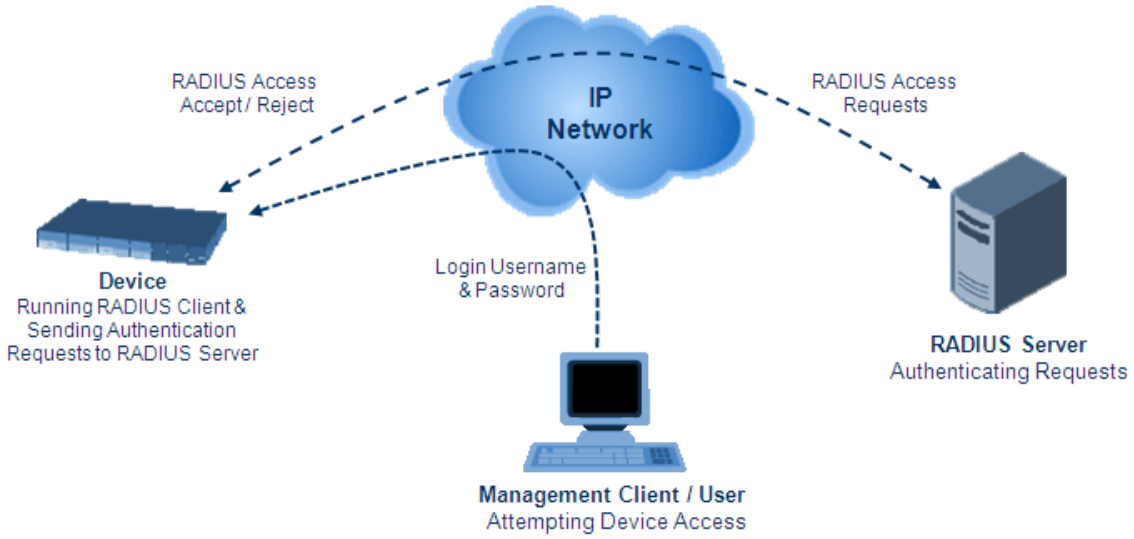
RADIUS-based Management User Authentication

You can enhance security for your device by implementing Remote Authentication Dial-In User Service (RADIUS per RFCs 2865) for authenticating multiple management user accounts of the device's embedded Web and Telnet (CLI) servers. Thus, RADIUS also prevents unauthorized access to your device.

When RADIUS authentication is not used, the user's login username and password are locally authenticated by the device using the Local Users table (see Configuring Management User Accounts). However, you can configure the device to use the Local Users table as a fallback mechanism if the RADIUS server does not respond.

> If you enable RADIUS-based user login authentication, when users with Security Administrator privilege level log in to the device's CLI, they are automatically given access to the CLI privileged mode ("#"). For all other user privilege levels, the user needs to run the **enable** command and then enter the password to access the CLI privileged mode.

When RADIUS authentication is used, the RADIUS server stores the user accounts - usernames, passwords, and access levels (authorization). When a management user (client) tries to access the device, the device sends the RADIUS server the user's username and password for authentication. The RADIUS server replies with an acceptance or a rejection notification. During the RADIUS authentication process, the device's Web interface is blocked until an acceptance response is received from the RADIUS server. Communication between the device and the RADIUS server is done using a shared secret, which is not transmitted over the network.



To implement RADIUS, you need to do the following:

- Set up a RADIUS server (third-party) to communicate with the device - see Setting Up a Third-Party RADIUS Server
- Configure the device as a RADIUS client for communication with the RADIUS server - see Configuring RADIUS Authentication

Setting Up a Third-Party RADIUS Server

The following procedure provides an example for setting up a third-party RADIUS sever, *FreeRADIUS* which can be downloaded from www.freeradius.org. Follow the instructions on this Web site for installing and configuring the server. If you use a RADIUS server from a different vendor, refer to its appropriate documentation.

➢ **To set up a third-party RADIUS server (e.g., *FreeRADIUS)*:**

1. Define the device as an authorized client of the RADIUS server, with the following:
   - Predefined *shared secret* (password used to secure communication between the device and the RADIUS server)
   - Vendor ID (configured on the device in Configuring the RADIUS Vendor ID)

   Below is an example of the *clients.conf* file (FreeRADIUS client configuration):

```
#
# clients.conf - client configuration directives
#
client 10.31.4.47 {
      secret        = FutureRADIUS
      shortname     = audc_device
}
```

2. If access levels are required, set up a Vendor-Specific Attributes (VSA) dictionary for the RADIUS server and select an attribute ID that represents each user's access level. The example below shows a dictionary file for FreeRADIUS that defines the attribute "ACL-Auth-Level" with "ID=35". For the device's user access levels and their corresponding numeric representation in RADIUS servers, see Configuring Management User Accounts.

```
#
# AudioCodes VSA dictionary
#
VENDOR AudioCodes 5003
ATTRIBUTE ACL-Auth-Level 35 integer AudioCodes
VALUE ACL-Auth-Level ACL-Auth-UserLevel 50
VALUE ACL-Auth-Level ACL-Auth-AdminLevel 100
VALUE ACL-Auth-Level ACL-Auth-SecurityAdminLevel 200
```

3. Define the list of users authorized to use the device, using one of the password authentication methods supported by the server implementation. The example below shows a user configuration file for FreeRADIUS using a plain-text password:

```
# users - local user configuration database
john    Auth-Type := Local, User-Password == "qwerty"
        Service-Type = Login-User,
        ACL-Auth-Level = ACL-Auth-SecurityAdminLevel
sue   Auth-Type := Local, User-Password == "123456"
        Service-Type = Login-User,
        ACL-Auth-Level = ACL-Auth-UserLevel
```

4. Record and retain the IP address, port number, shared secret code, vendor ID, and VSA access level identifier (if access levels are implemented) used by the RADIUS server.

Configuring RADIUS-based User Authentication

The following procedure describes how to configure RADIUS-based login authentication. For a detailed description of the RADIUS parameters, see RADIUS Parameters.

➢ **To configure RADIUS-based login authentication:**

1. Open the Authentication Server page (**Setup** menu > **Administration** tab > **Web & CLI** folder > **Authentication Server**).
2. From the 'Use RADIUS for Web/Telnet Login' drop-down list, select **Enable** to enable RADIUS authentication for Web and Telnet login:

3. When implementing Web user access levels, do one of the following:

- **If the RADIUS server response includes the access level attribute:** In the 'RADIUS VSA Access Level Attribute' field, enter the code that indicates the access level attribute in the VSA section of the received RADIUS packet. For defining the RADIUS server with access levels, see Setting Up a Third-Party RADIUS Server.

- **If the RADIUS server response does not include the access level attribute:** In the 'Default Access Level' field, enter the default access level that is applied to all users authenticated by the RADIUS server.

4. Configure RADIUS timeout handling:
   a. From the 'Behavior upon Authentication Server Timeout' drop-down list, select the option if the RADIUS server does not respond within five seconds:
      - **Deny Access**: device denies user login access.
      - **Verify Access Locally**: device checks the username and password configured locally for the user in the Local Users table (see Configuring Management User Accounts), and if correct, allows access.
   b. In the 'Password Local Cache Timeout' field, enter a time limit (in seconds) after which the username and password verified by the RADIUS server becomes invalid and a username and password needs to be re-validated with the RADIUS server.
   c. From the 'Password Local Cache Mode' drop-down list, select the option for the local RADIUS password cache timer:
      - **Reset Timer Upon Access**: upon each access to a Web page, the timer resets (reverts to the initial value configured in the previous step).
      - **Absolute Expiry Timer**: when you access a Web page, the timer doesn't reset, but continues its count down.

5. Configure when the Local Users table must be used to authenticate login users. From the 'Use Local Users Database' drop-down list, select one of the following:
   - **When No Auth Server Defined (default):** When no RADIUS server is configured or if a server is configured but connectivity with the server is down (if the server is up, the device authenticates the user with the server).
   - **Always:** First attempts to authenticate the user using the Local Users table, but if not found, it authenticates the user with the RADIUS server.

6. Click **Apply**, and then reset the device with a save-to-flash for your settings to take effect.

Securing RADIUS Communication

RADIUS authentication requires HTTP basic authentication (according to RFC 2617). However, this is insecure as the usernames and passwords are transmitted in clear text over plain HTTP. Thus, as digest authentication is not supported with RADIUS, it is recommended that you use HTTPS with RADIUS so that the usernames and passwords are encrypted. To enable the device to use HTTPS, configure the 'Secured Web Connection (HTTPS)' parameter to **HTTPS Only** (see Configuring Secured (HTTPS) Web).

RADIUS-based User Authentication in URL

RADIUS authentication of the management user is typically done after the user accesses the Web interface by entering only the device's IP address in the Web browser's URL field (for example, http://10.13.4.12/) and then entering the username and password credentials in the Web interface's login screen. However, authentication with the RADIUS server can also be done immediately after the user enters the URL, if the URL also contains the login credentials. For example: http://10.4.4.112/Forms/RadiusAuthentication?WSBackUserName=John&WSBackPassword=1234.

This feature allows up to five simultaneous users only.

RADIUS-based CDR Accounting

Once you have configured a RADIUS server(s) for accounting in Configuring RADIUS Servers, you need to enable and configure RADIUS- based CDR accounting (see Configuring RADIUS Accounting).

LDAP-based Management and SIP Services

The device supports the Lightweight Directory Access Protocol (LDAP) application protocol and can operate with third-party, LDAP-compliant servers such as Microsoft Active Directory (AD).

You can use LDAP for the following LDAP services:

- **SIP-related (Control) LDAP Queries:** LDAP can be used for routing and manipulation (e.g., calling name and destination address).

  The device connects and binds to the remote LDAP server (IP address or DNS/FQDN) during the service's initialization (at device start-up) or whenever you change the LDAP server's IP address and port. Binding to the LDAP server is based on username and password (Bind DN and Password). Service makes 10 attempts to connect and bind to the remote LDAP server, with a timeout of 20 seconds between attempts. If connection fails, the service remains in disconnected state until the LDAP server's IP address or port is changed. If connection to the LDAP server later fails, the service attempts to reconnect.

  For the device to run a search, the path to the directory's subtree, known as the distinguished name (DN), where the search is to be done must be configured (see Configuring LDAP DNs (Base Paths) per LDAP Server). The search key (filter), which defines

the exact DN to search and one or more attributes whose values must be returned to the device must also be configured. For more information on configuring these attributes and search filters, see AD-based Routing for Microsoft Skype for Business.

The device can store recent LDAP queries and responses in its local cache. The cache is used for subsequent queries and/or in case of LDAP server failure. For more information, see Configuring the Device's LDAP Cache.

If connection with the LDAP server disconnects (broken), the device sends the SNMP alarm, acLDAPLostConnection. Upon successful reconnection, the alarm clears. If connection with the LDAP server is disrupted during the search, all search requests are dropped and an alarm indicating a failed status is sent to client applications.

- **Management-related LDAP Queries:** LDAP can be used for authenticating and authorizing management users (Web and CLI) and is based on the user's login username and password (credentials) when attempting login to one of the device's management platforms. When configuring the login username (LDAP Bind DN) and password (LDAP Password) to send to the LDAP server, you can use templates based on the dollar ($) sign, which the device replaces with the actual username and password entered by the user during the login attempt. You can also configure the device to send the username and password in clear-text format or encrypted using TLS (SSL).

  The device connects to the LDAP server (i.e., an LDAP session is created) only when a login attempt occurs. The LDAP Bind operation establishes the authentication of the user based on the username-password combination. The server typically checks the password against the userPassword attribute in the named entry. A successful Bind operation indicates that the username-password combination is correct; a failed Bind operation indicates that the username-password combination is incorrect.

  Once the user is successfully authenticated, the established LDAP session may be used for further LDAP queries to determine the user's management access level and privileges (Operator, Admin, or Security Admin). This is known as the user authorization stage. To determine the access level, the device searches the LDAP directory for groups of which the user is a member, for example:

  ```
  CN=\# Support Dept,OU=R&D Groups,OU=Groups,OU=APC,OU=Japan,OU=ABC,DC=corp,DC=abc,DC=com
  CN=\#AllCellular,OU=Groups,OU=APC,OU=Japan,OU=ABC,DC=corp,DC=abc,DC=com
  ```

  The device then assigns the user the access level configured for that group (in Configuring Access Level per Management Groups Attributes). The location in the directory where you want to search for the user's member group(s) is configured using the following:

  - Search base object (distinguished name or DN, e.g., "ou=ABC,dc=corp,dc=abc,dc=com"), which defines the location in the directory from where the LDAP search begins and is configured in Configuring LDAP DNs (Base Paths) per LDAP Server.

- Search filter, for example, (&(objectClass=person)(sAMAccountName=JohnD)), which filters the search in the subtree to include only the specific username. The search filter can be configured with the dollar ($) sign to represent the username, for example, (sAMAccountName=$). To configure the search filter, see Configuring the LDAP Search Filter Attribute.
- Management attribute (e.g., memberOf), from where objects that match the search filter criteria are returned. This shows the user's member groups. The attribute is configured in the LDAP Servers table (see Configuring LDAP Servers).

If the device finds a group, it assigns the user the corresponding access level and permits login; otherwise, login is denied. Once the LDAP response has been received (success or failure), the device ends the LDAP session.

- **LDAP-based Management services:** This LDAP service works together with the LDAP-based management account (described above), allowing you to use different LDAP service accounts for user authentication and user authorization:
  - Management-type LDAP server: This LDAP server account is used only for user authentication. For more information about how it works, see Management-related LDAP Queries, above.
  - Management Service-type LDAP server: This LDAP server account is used only for user authorization (i.e., the user's management access level and privileges). The device has an always-on connection with the LDAP server and uses a configured (fixed) LDAP username (Bind Name) and password. Only if user authentication succeeds, does the device query this Management Service-type LDAP server account for user authorization. Thus, management groups and DNs are configured only for this LDAP server account (instead of for the regular LDAP-based management account).

  Therefore, user authorization is done only by a specific LDAP "administrator", which has a fixed username and password. In contrast, user authentication is done by the user itself (i.e., binding to the LDAP account with each user's username and password). Having a dedicated LDAP account for user authorization may provide additional security to the network by preventing users from accessing the authorization settings in the LDAP server.

For all the previously discussed LDAP services, the following additional LDAP functionality is supported:

- Search method for searching DN object records between LDAP servers and within each LDAP server (see Configuring LDAP Search Methods).
- Default access level that is assigned to the user if the queried response does not contain an access level.
- Local Users table for authenticating users instead of the LDAP server (for example, when a communication problem occurs with the server). For more information, see Configuring Local Database for Management User Authentication.

Enabling the LDAP Service

Before you can configure LDAP support, you need to enable the LDAP service.

➢ **To enable LDAP:**

1. Open the LDAP Settings page (**Setup** menu > **IP Network** tab > **RADIUS & LDAP** folder > **LDAP Settings**).

2. From the 'LDAP Service' drop-down list, select **Enable**.
3. Click **Apply**, and then reset the device with a save-to-flash for your settings to take effect.

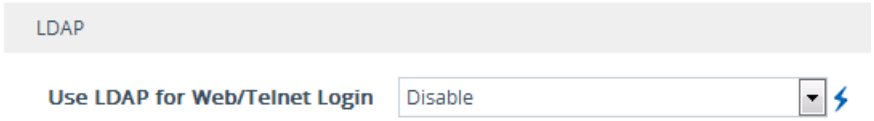
Enabling LDAP-based Web/CLI User Login Authentication and Authorization

The LDAP service can be used for authenticating and authorizing device management users (Web and CLI) based on the user's login username and password (credentials). At the same, it can also be used to determine users' management access levels (privileges). Before you can configure LDAP-based login authentication, you must enable this type of LDAP service.

> If you enable LDAP-based user login authentication, when users with Security Administrator privilege level log in to the device's CLI, they are automatically given access to the CLI privileged mode ("#"). For all other user privilege levels, the user needs to run the **enable** command and then enter the password to access the CLI privileged mode.

➢ **To enable LDAP-based login authentication:**

1. Open the Authentication Server page (**Setup** menu > **Administration** tab > **Web & CLI** folder > **Authentication Server**).



2. Under the LDAP group, from the 'Use LDAP for Web/Telnet Login' drop-down list, select **Enable**.
3. Click **Apply**, and then reset the device with a save-to-flash for your settings to take effect.

Configuring LDAP Server Groups

The LDAP Server Groups table lets you configure up to 250 LDAP Server Groups. An LDAP Server Group is a logical configuration entity that contains up to two LDAP servers. LDAP servers are assigned to LDAP Server Groups in the LDAP Servers table (see Configuring LDAP Servers). To use a configured LDAP server, you must assign it to an LDAP Server Group. You can configure the following types of LDAP Server Groups (configured by the 'Type' parameter described below):

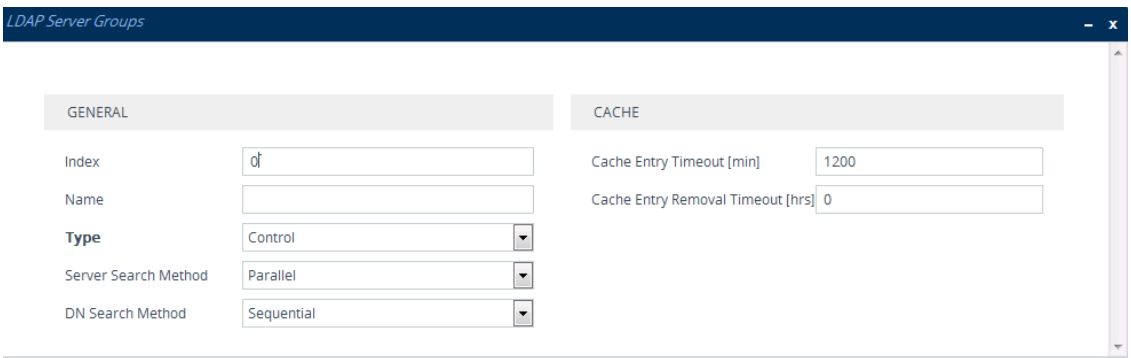
- **Control:** To use an LDAP server for call routing, you need to configure the LDAP Server Group as a **Control** type, and then assign the LDAP Server Group to a Routing Policy. The Routing Policy in turn needs to be assigned to the relevant routing rule(s). You can assign a Routing Policy to only one LDAP Server Group. Therefore, for multi-tenant deployments where multiple Routing Policies are employed, each tenant can be assigned a specific LDAP Server Group through its unique Routing Policy.
- **Management:** To use an LDAP server for management where it does user login authentication and user authorization, you need to configure the LDAP Server Group as a **Management** type. Additional LDAP-based management parameters need to be configured, as described in Enabling LDAP-based Web/CLI User Login Authentication and Authorization and Configuring LDAP Servers.
- **Management Service**: To use two different LDAP server accounts for management where one LDAP account does user authentication and the other LDAP account does user authorization, you need to configure two LDAP Server Groups. Configure the LDAP Server Group for user authentication as a **Management** type and the LDAP Server Group for user authorization as a **Management Service** type. In this setup, configure all the user-authorization settings (i.e., Management LDAP Groups and LDAP Server Search Base DN) only for the Management Service-type LDAP Server Group (instead of for the Management-type LDAP Server Group).

The following procedure describes how to configure an LDAP Server Group through the Web interface. You can also configure it through ini file [LDAPServerGroups] or CLI (`configure system > ldap ldap-server-groups`).

> The device provides a preconfigured LDAP Server Group ("DefaultCTRLServersGroupin") in the LDAP Server Groups table, which can be modified or deleted.

➢ To configure an LDAP Server Group:

1. Open the LDAP Server Groups table (**Setup** menu > **IP Network** tab > **RADIUS & LDAP** folder > **LDAP Server Groups**).
2. Click **New**; the following dialog box appears:



3. Configure an LDAP Server Group according to the parameters described in the table below.

4. Click **Apply**.

Table 16-8: LDAP Server Groups Table Parameter Descriptions

Parameter	Description
General	
'Index' [LdapServerGroups_Index]	Defines an index number for the new table row. **Note:** Each row must be configured with a unique index.
'Name' `name` [LdapServerGroups_Name]	Defines a descriptive name, which is used when associating the row in other tables. The valid value is a string of up to 20 characters. **Note:** ■ Each row must be configured with a unique name. ■ The parameter value cannot contain a forward slash (/).
'Type' `server-type` [LdapServerGroups_ ServerType]	Defines whether the servers in the group are used for SIP-related LDAP queries (Control) or management login authentication-related LDAP queries (Management). ■ [0] **Control** (default) ■ [1] **Management** ■ [2] **Management Service** For more information on the different optional LDAP services, see LDAP-based Management and SIP Services on page 276. **Note:** ■ For table row Index #0, the parameter can only be configured to **Control**. ■ Only one LDAP Server Group can be configured for management.
'Server Search Method' `server-search-method` [LdapServerGroups_ SearchMethod]	Defines the method for querying between the two LDAP servers in the group. ■ [0] **Parallel** = (Default) The device queries the LDAP servers at the same time. ■ [1] **Sequential** = The device first queries one of the LDAP servers and if the DN object is not found or the search fails, it queries the second LDAP server.

Parameter	Description
'DN Search Method' `search-dn-method` [LdapServerGroups_ SearchDnsMethod]	Defines the method for querying the Distinguished Name (DN) objects within each LDAP server. ■ [0] **Sequential** = (Default) The query is done in each DN object, one by one, until a result is returned. For example, a search for the DN object record "JohnD" is first run in DN object "Marketing" and if a result is not found, it searches in "Sales", and if not found, it searches in "Administration", and so on. ■ [1] **Parallel** = The query is done in all DN objects at the same time. For example, a search for the DN object record "JohnD" is done at the same time in the "Marketing", "Sales" and "Administration" DN objects.
Cache	
'Cache Entry Timeout' `cache-entry-timeout` [LdapServersGroups_ CacheEntryTimeout]	Defines the duration (in minutes) that an entry in the device's LDAP cache is valid. If the timeout expires, the cached entry is used only if there is no connectivity with the LDAP server. The valid range is 0 to 35791. The default is 1200. If set to 0, the LDAP entry is always valid.
'Cache Entry Removal Timeout' `cache-entry-removal-timeout` [LdapServerGroups_ CacheEntryRemovalTimeout]	Defines the duration (in hours) after which the LDAP entry is deleted from the device's LDAP cache. The valid range is 0 to 596. The default is 0 (i.e., the entry is never deleted).

Configuring LDAP Servers

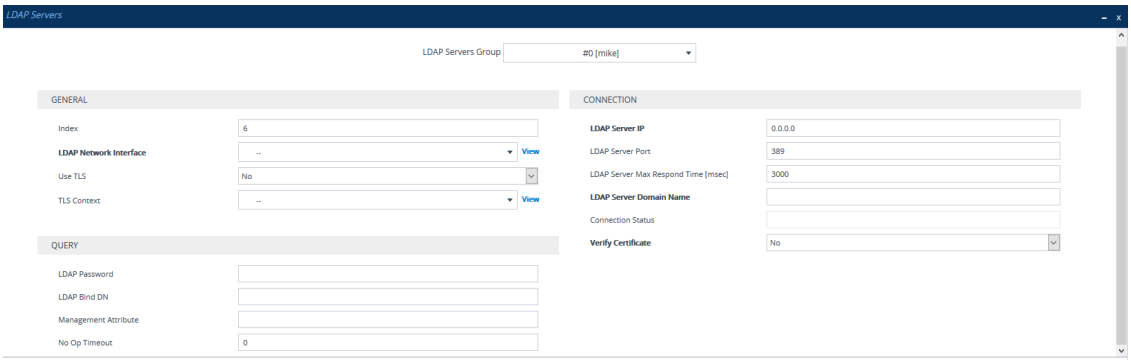
The LDAP Servers table lets you configure up to 500 LDAP servers. The table defines the address and connectivity settings of the LDAP server. The LDAP server can be configured for SIP-related queries (e.g., routing and manipulation) or LDAP-based management user login authentication and authorization (username-password).

The following procedure describes how to configure an LDAP server through the Web interface. You can also configure it through ini file [LdapConfiguration] or CLI (`configure system > ldap ldap-configuration`).

When you configure an LDAP server, you need to assign it an LDAP Server Group. Therefore, before you can configure an LDAP server in the table, you must first configure at least one LDAP Server Group in the LDAP Server Groups table (see Configuring LDAP Server Groups).

➢ To configure an LDAP server:

1. Open the LDAP Servers table (**Setup** menu > **IP Network** tab > **RADIUS & LDAP** folder > **LDAP Servers**).
2. Click **New**; the following dialog box appears:



3. Configure an LDAP server according to the parameters described in the table below.
4. Click **Apply**.

Table 16-9: LDAP Servers Table Parameter Descriptions

Parameter	Description
General	
'Index' [LdapConfiguration_Index]	Defines an index number for the new table row. **Note:** Each row must be configured with a unique index.
'LDAP Servers Group' `server-group` [LdapConfiguration_Group]	Assigns the LDAP server to an LDAP Server Group, configured in the LDAP Server Groups table (see Configuring LDAP Server Groups). **Note:** ■ The parameter is mandatory and must be set before configuring the other parameters in the table. ■ Up to two LDAP servers can be assigned to the same LDAP Server Group.
'LDAP Network Interface'	Assigns one of the device's IP network

Parameter	Description
interface-type [LdapConfiguration_Interface]	interfacesthrough which communication with the LDAP server is done. By default, no value is defined and the device uses the OAMP network interface, configured in the IP Interfaces table. To configure IP network interfaces, see Configuring IP Network Interfaces. **Note:** The parameter is mandatory.
'Use TLS' use-tls [LdapConfiguration_useTLS]	Enables the device to encrypt the username and password (for Control and Management related queries) using TLS when sending them to the LDAP server. ■ [0] **No** = (Default) Username and password are sent in clear-text format. ■ [1] **Yes**
'TLS Context' tls-context [LdapConfiguration_ ContextName]	Assigns a TLS Context (TLS configuration) for the connection with the LDAP server. By default, no value is defined and the device uses the default TLS Context (ID 0). To configure TLS Contexts, see Configuring TLS Certificates on page 158. **Note:** The parameter is applicable only if the 'Use TLS' parameter is configured to **Yes**.
Connection	
'LDAP Server IP' server-ip [LdapConfiguration_ LdapConfServerIp]	Defines the IP address of the LDAP server (in dotted-decimal notation, e.g., 192.10.1.255). By default, no IP address is defined. **Note:** ■ The parameter is mandatory. ■ If you want to use an FQDN for the LDAP server, leave the parameter undefined and configure the FQDN in the 'LDAP Server Domain Name' parameter (see below).
'LDAP Server Port' server-port	Defines the port number of the LDAP server. The valid value range is 0 to 65535. The default port

Parameter	Description
[LdapConfiguration_ LdapConfServerPort]	number is 389.
'LDAP Server Max Respond Time' `max-respond-time` [LdapConfiguration_ LdapConfServerMaxRespondTime]	Defines the duration (in msec) that the device waits for LDAP server responses. The valid value range is 0 to 86400. The default is 3000. **Note:** ■ If the response time expires, you can configure the device to use the Local Users table for authenticating the user. For more information, see Configuring Local Database for Management User Authentication. ■ Activation of this timeout depends on connection type: ✓ Normal TCP connection: The device starts the timer when it sends the LDAP request. If no response is received from the LDAP server within the configured time, the device closes the connection. ✓ TLS connection: The device first performs the TLS handshake and once negotiation completes, it sends the LDAP request. The device starts the timer only from the first TLS message sent during the handshake (and not from the LDAP request).
'LDAP Server Domain Name' `domain-name` [LdapConfiguration_ LdapConfServerDomainName]	Defines the domain name (FQDN) of the LDAP server. The device tries to connect to the LDAP server according to the IP address listed in the received DNS query. If there is no connection to the LDAP server or the connection to the LDAP server fails, the device tries to connect to the LDAP server with the next IP address in the DNS query list. **Note:** If the 'LDAP Server IP' parameter is configured, the 'LDAP Server Domain Name' parameter is ignored. Thus, if you want to use an FQDN, leave the 'LDAP Server IP' parameter undefined.
'Verify Certificate' `verify-certificate`	Enables certificate verification when the connection with the LDAP server uses TLS.

Parameter	Description
[LdapConfiguration_ VerifyCertificate]	■ [0] **No** = (Default) No certificate verification is done. ■ [1] **Yes** = The device verifies the authentication of the certificate received from the LDAP server. The device authenticates the certificate against the trusted root certificate store associated with the associated TLS Context (see 'TLS Context' parameter above) and if ok, allows communication with the LDAP server. If authentication fails, the device denies communication (i.e., handshake fails). The device can also authenticate the certificate by querying with an Online Certificate Status Protocol (OCSP) server whether the certificate has been revoked. This is also configured for the associated TLS Context. **Note:** The parameter is applicable only if the 'Use TLS' parameter is configured to **Yes**.
'Verify Certificate Subject Name' `verify-subject-Name` [LdapConfiguration_ VerifySubjectName]	Enables the verification of the TLS certificate subject name (Common Name / CN or Subject Alternative Name / SAN) that is used in the incoming connection request from the LDAP server. ■ [0] **Disable** = (Default) No verification is done. ■ [1] **Enable** = The device verifies the subject name of the certificate received from the LDAP server with the hostname or IP address configured for the LDAP server. If authentication fails, the device denies communication (i.e., handshake fails). **Note:** The parameter is applicable only if the 'Use TLS' parameter is configured to **Yes**.
'Connection Status' `connection-status` [LdapConfiguration_ ConnectionStatus]	(Read-only) Displays the connection status with the LDAP server. ■ "Not Applicable" ■ "LDAP Connection Broken" ■ "Connecting" ■ "Connected" For more information about a disconnected LDAP

Parameter	Description
	connection, see your Syslog messages generated by the device.
Query	
'LDAP Password' password [LdapConfiguration_ LdapConfPassword]	Defines the user password for accessing the LDAP server during connection and binding operations. ■ LDAP-based SIP queries: The parameter is the password used by the device to authenticate itself, as a client, to obtain LDAP service from the LDAP server. ■ LDAP-based user login authentication: The parameter represents the login password entered by the user during a login attempt. You can use the $ (dollar) sign in this value to enable the device to automatically replace the $ sign with the user's login password in the search filter, which it sends to the LDAP server for authenticating the user's username-password combination. For example, $. **Note:** ■ The parameter is mandatory. ■ By default, the device sends the password in clear-text format. You can enable the device to encrypt the password using TLS (see the 'Use TLS' parameter in this table). ■ The password cannot be configured with wide characters.
'LDAP Bind DN' bind-dn [LdapConfiguration_ LdapConfBindDn]	Defines the LDAP server's bind Distinguished Name (DN) or username. ■ LDAP-based SIP queries: The DN is used as the username during connection and binding to the LDAP server. The DN is used to uniquely name an AD object. Below are example parameter settings: ✓ cn=administrator,cn=Users,dc=domain,dc=com

Parameter	Description
	✓ administrator@domain.com ✓ domain\administrator ■ LDAP-based user login authentication: The parameter represents the login username entered by the user during a login attempt. You can use the $ (dollar) sign in this value to enable the device to automatically replace the $ sign with the user's login username in the search filter, which it sends to the LDAP server for authenticating the user's username-password combination. An example configuration for the parameter is $@sales.local, where the device replaces the $ with the entered username, for example, JohnD@sales.local. The username can also be configured with the domain name of the LDAP server. **Note:** By default, the device sends the username in clear-text format. You can enable the device to encrypt the username using TLS (see the 'Use TLS' parameter in this table).
'Management Attribute' `mgmt-attr` [LdapConfiguration_ MngmAuthAtt]	Defines the LDAP attribute name to query, which contains a list of groups to which the user is a member. For Active Directory, this attribute is typically "memberOf". The attribute's values (groups) are used to determine the user's management access level; the group's corresponding access level is configured in Configuring Access Level per Management Groups Attributes. **Note:** ■ The parameter is applicable only to LDAP-based login authentication and authorization (i.e., the 'Type' parameter is set to **Management**). ■ If this functionality is not used, the device assigns the user the configured default access level. For more information, see Configuring Access Level per Management Groups Attributes.
'No Op Timeout' `noop-timeout`	Defines the timeout (in minutes) of inactivity in the connection between the device and the LDAP server, after which the device sends an LDAP "abandon"

Parameter	Description
[LdapConfiguration_ NoOpTimeout]	request to keep the LDAP connection alive (i.e., LDAP persistent connection). The valid value to enable this feature is any value greater than 0. The default is 0 (i.e., if there is no activity on the connection, the device does not send "abandon" requests and the LDAP server may disconnect). **Note:** The parameter is applicable only to LDAP connections that are used for routing (i.e., the 'Type' parameter is configured to **Control**).

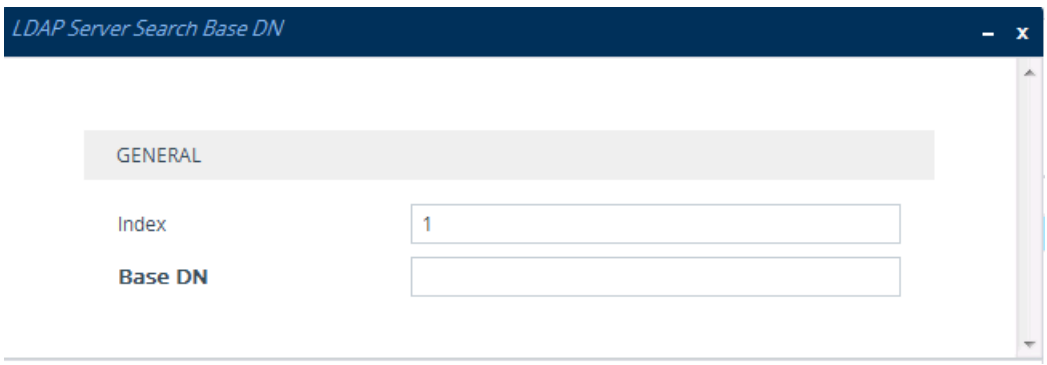
Configuring LDAP DNs (Base Paths) per LDAP Server

The LDAP Search DN table lets you configure LDAP base paths. The table is a "child" of the LDAP Servers table (see Configuring LDAP Servers) and configuration is done per LDAP server. For the device to run a search using the LDAP service, the base path to the directory's subtree, referred to as the distinguished name object (or DN), where the search is to be done must be configured. For each LDAP server, you can configure up to three base paths.

The following procedure describes how to configure DNs per LDAP server through the Web interface. You can also configure it through ini file [LdapServersSearchDNs] or CLI (`configure system > ldap ldap-servers-search-dns`).

➢ **To configure an LDAP base path per LDAP server:**

1. Open the LDAP Servers table (**Setup** menu > **IP Network** tab > **RADIUS & LDAP** folder > **LDAP Servers**).
2. In the table, select the row of the LDAP server for which you want to configure DN base paths, and then click the **LDAP Servers Search Based DNs** link located below the table; the LDAP Server Search Base DN table opens.
3. Click **New**; the following dialog box appears:



4. Configure an LDAP DN base path according to the parameters described in the table below.

5. Click **Apply**, and then save your settings to flash memory.

Table 16-10:LDAP Server Search Base DN Table Parameter Descriptions

Parameter	Description
'Index' `set internal-index` [LdapServersSearchDNs_Index]	Defines an index number for the new table row. **Note:** Each row must be configured with a unique index.
'Base DN' `set base-path` [LdapServersSearchDNs_Base_Path]	Defines the full path (DN) to the objects in the AD where the query is done. The valid value is a string of up to 256 characters. For example: OU=NY,DC=OCSR2,DC=local. In this example, the DN path is defined by the LDAP names, OU (organizational unit) and DC (domain component).

Configuring the LDAP Search Filter Attribute

When the LDAP-based login username-password authentication succeeds, the device searches the LDAP server for all groups of which the user is a member. The LDAP query is based on the following LDAP data structure:

- **Search base object (distinguished name or DN, e.g., "ou=ABC,dc=corp,dc=abc,dc=com"):** The DN defines the location in the directory from which the LDAP search begins and is configured in Configuring LDAP DNs (Base Paths) per LDAP Server.
- **Filter (e.g., "(&(objectClass=person)(sAMAccountName=johnd))"):** This filters the search in the subtree to include only the login username (and excludes others). This is configured by the 'LDAP Authentication Filter' parameter, as described in the following procedure. You can use the dollar ($) sign to represent the username. For example, the filter can be configured as "(sAMAccountName=$)", where if the user attempts to log in with the username "SueM", the LDAP search is done only for the attribute sAMAccountName that equals "SueM".
- **Attribute (e.g., "memberOf") to return from objects that match the filter criteria:** The attribute is configured by the 'Management Attribute' parameter in the LDAP Servers table (see Configuring LDAP Servers).

Therefore, the LDAP response includes only the groups of which the specific user is a member.

- The search filter is applicable only to LDAP-based login authentication and authorization queries.
- The search filter is a global setting that applies to all LDAP-based login authentication and authorization queries, across all configured LDAP servers.

➢ **To configure the LDAP search filter for management users:**

1. Open the LDAP Settings page (**Setup** menu > **IP Network** tab > **RADIUS & LDAP** folder > **LDAP Settings**).
2. In the 'LDAP Authentication Filter' field, enter the LDAP search filter attribute for searching the login username for user authentication:

3. Click **Apply**.

Configuring Access Level per Management Groups Attributes

The Management LDAP Groups table lets you configure LDAP group objects and their corresponding management user access level. The table is a "child" of the LDAP Servers table (see Configuring LDAP Servers) and configuration is done per LDAP server. For each LDAP server, you can configure up to three row entries of LDAP group(s) with their corresponding access level (only one row for each level).

- The Management LDAP Groups table is applicable only to LDAP-based login authentication and authorization queries.
- If the LDAP response received by the device includes multiple groups of which the user is a member and you have configured different access levels for some of these groups, the device assigns the user the highest access level. For example, if the user is a member of two groups where one has access level **Monitor** and the other **Admin**, the device assigns the user the **Admin** access level.
- When the access level is unknown, the device assigns the default access level to the user, configured by the 'Default Access Level' parameter as used also for RADIUS (see Configuring RADIUS-based User Authentication). This can occur in the following scenarios:
  - ✓ The user is not a member of any LDAP group.
  - ✓ The group of which the user is a member is not configured on the device (as described in this section).
  - ✓ The device is not configured to query the LDAP server for a management attribute (see Configuring LDAP Servers).

Group objects represent groups in the LDAP server of which the user is a member. The access level represents the user account's permissions and rights in the device's management interface (e.g., Web and CLI). The access level can either be **Monitor**, **Admin**, or **Security Admin**. For an explanation on the privileges of each level, see Configuring Management User Accounts.

When the username-password authentication with the LDAP server succeeds, the device searches the LDAP server for all groups of which the user is a member. The LDAP query is based on the following LDAP data structure:

- Search base object (distinguished name or DN, e.g., "ou=ABC,dc=corp,dc=abc,dc=com"), which defines the location in the directory from which the LDAP search begins. This is configured in Configuring LDAP DNs (Base Paths) per LDAP Server.
- Filter (e.g., "(&(objectClass=person)(sAMAccountName=johnd))"), which filters the search in the subtree to include only the login username (and excludes others). For configuration, see Configuring the LDAP Search Filter Attribute.
- Attribute (e.g., "memberOf") to return from objects that match the filter criteria. This attribute is configured by the 'Management Attribute' parameter in the LDAP Servers table.

The LDAP response includes all the groups of which the specific user is a member, for example:

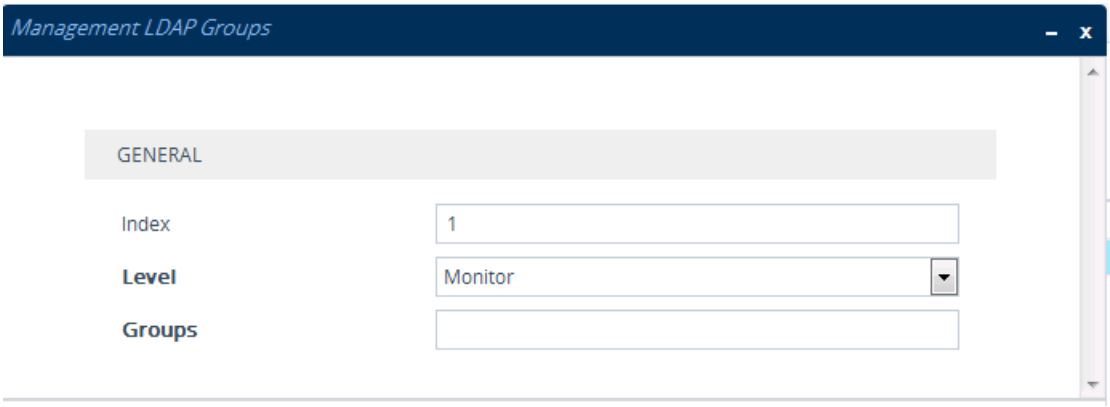
CN=\# Support Dept,OU=R&D Groups,OU-
U=Groups,OU=APC,OU=Japan,OU=ABC,DC=corp,DC=abc,DC=com
CN=\#AllCellular,OU=Groups,OU=APC,OU=Japan,OU=ABC,DC=corp,DC=abc,DC=com

The device searches this LDAP response for the group names that you configured in the Management LDAP Groups table in order to determine the user's access level. If the device finds a group name, the user is assigned the corresponding access level and login is permitted; otherwise, login is denied. Once the LDAP response has been received (success or failure), the LDAP session terminates.

The following procedure describes how to configure an access level per management groups through the Web interface. You can also configure it through ini file [MgmntLDAPGroups] or CLI (`configure system > ldap mgmt-ldap-groups`).

➢ **To configure management groups and corresponding access level:**

1. Open the LDAP Servers table (**Setup** menu > **IP Network** tab > **RADIUS & LDAP** folder > **LDAP Servers**).
2. In the table, select the row of the LDAP server for which you want to configure management groups with a corresponding access level, and then click the **Management LDAP Groups** link located below the table; the Management LDAP Groups table opens.
3. Click **New**; the following dialog box appears:



4. Configure a group name(s) with a corresponding access level according to the parameters described in the table below.

5. Click **Apply**, and then save your settings to flash memory.

Table 16-11:Management LDAP Groups Table Parameter Descriptions

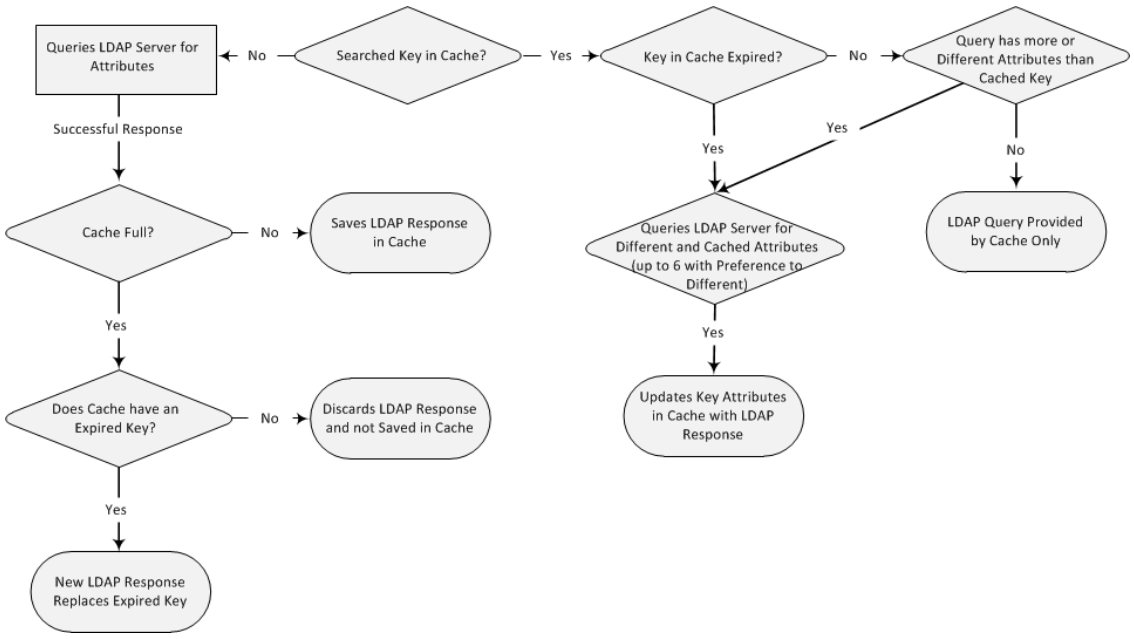
Parameter	Description
'Index' [MgmntLDAPGroups_GroupIndex]	Defines an index number for the new table row. **Note:** Each row must be configured with a unique index.
'Level' `level` [MgmntLDAPGroups_Level]	Defines the access level of the group(s). ■ [0] **Monitor** (Default) ■ [1] **Admin** ■ [2] **Security Admin** **Note:** You can configure only one row per access level.
'Groups' `groups` [MgmntLDAPGroups_Group]	Defines the attribute names of the groups in the LDAP server. The valid value is a string of up to 256 characters. To define multiple groups, separate each group name with a semicolon (;).

Configuring the Device's LDAP Cache

The device can optionally store LDAP queries of LDAP Attributes for a searched key with an LDAP server and the responses (results) in its local cache. The cache is used for subsequent queries and/or in case of LDAP server failure. The benefits of this feature include the following:

- Improves routing decision performance by using local cache for subsequent LDAP queries
- Reduces number of queries performed on an LDAP server and corresponding bandwidth consumption
- Provides partial survivability in case of intermittent LDAP server failure (or network isolation)

The handling of LDAP queries using the device's LDAP cache is shown in the flowchart below:



If an LDAP query is required for an Attribute of a key that is already cached with that same Attribute, instead of sending a query to the LDAP server, the device uses the cache. However, if an LDAP query is required for an Attribute that does not appear for the cached key, the device queries the LDAP server, and then saves the new Attribute (and response) in the cache for that key.

If the device queries the LDAP server for different Attributes for a cached key, the device also includes already cached Attributes of the key, while adhering to the maximum number of allowed saved Attributes (see note below), with preference to the different Attributes. In other words, if the cached key already contains the maximum Attributes and an LDAP query is required for a different Attribute, the device sends an LDAP query to the server for the different Attribute and for the five **most recent** Attributes already cached with the key. Upon the LDAP response, the new Attribute replaces the **oldest** cached Attribute while the values of the other Attributes are refreshed with the new response.

The following table shows an example of different scenarios of LDAP queries of a cached key whose cached Attributes include *a*, *b* , *c*, and *d*, where *a* is the oldest and *d* the most recent Attribute:

Table 16-12:Example of LDAP Query for Cached Attributes

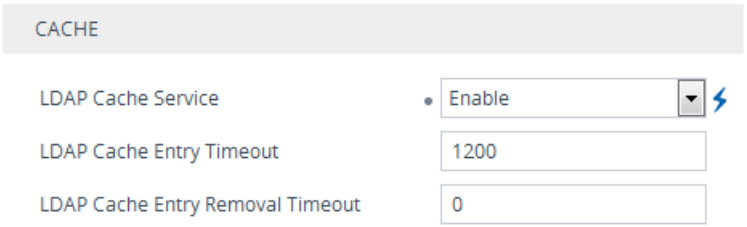
Attributes Requested in New LDAP Query for Cached Key	Attributes Sent in LDAP Query to LDAP Server	Attributes Saved in Cache after LDAP Response
e	**e**, a, b, c, d	**e**, a, b, c, d
e, **f**	**e**, **f**, a, b, c, d	**e**, **f**, a, b, c, d
e, **f**, **g**, **h**,**i**	**e**, **f**, **g**, **h**, **i**, d	**e**, **f**, **g**, **h**,**i**, d
e, **f**, **g**, **h**, **i**, **j**	**e**, **f**, **g**, **h**, **i**, **j**	**e**, **f**, **g**, **h**, **i**, **j**

- The LDAP Cache feature is applicable only to LDAP-based SIP queries (Control).
- The maximum LDAP cache size is 10,000 entries.
- The device can save up to six LDAP Attributes in the cache per searched LDAP key.
- The device also saves in the cache queried Attributes that do not have any values in the LDAP server.

The following procedure describes how to configure the device's LDAP cache through the Web interface. For a full description of the cache parameters, see LDAP Parameters.

➢ **To enable and configure the LDAP cache:**

1. Open the LDAP Settings page (**Setup** menu > **IP Network** tab > **RADIUS & LDAP** folder > **LDAP Settings**).

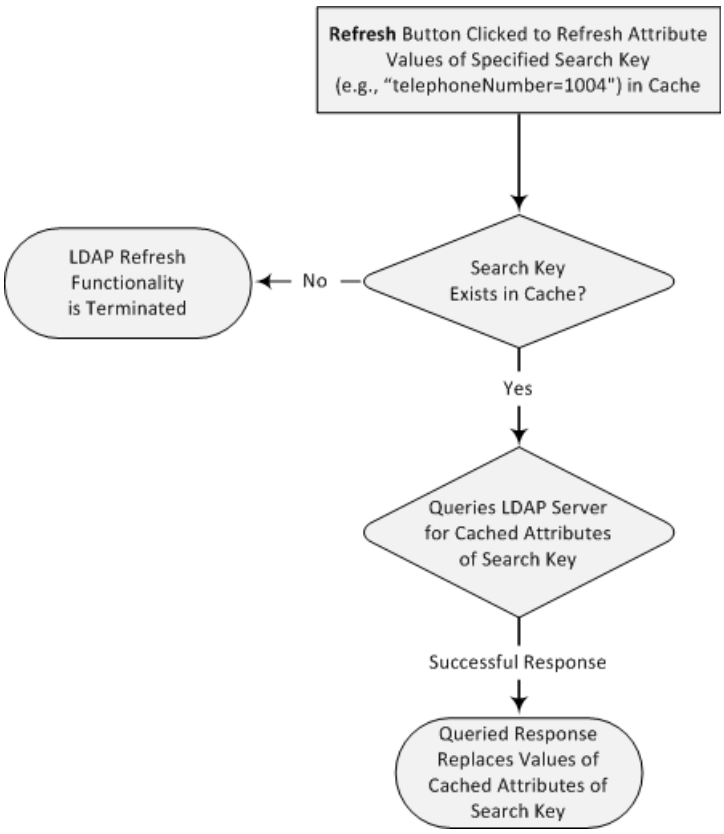


2. From the 'LDAP Cache Service' drop-down list, select **Enable** to enable LDAP cache.
3. In the 'LDAP Cache Entry Timeout' field, enter the duration (in minutes) for which an entry in the LDAP cache is valid.
4. In the 'LDAP Cache Entry Removal Timeout' field, enter the duration (in hours) after which the device removes the LDAP entry from the cache.
5. Click **Apply**, and then reset the device with a save-to-flash for your settings to take effect.

Refreshing the LDAP Cache

You can refresh values of LDAP Attributes associated with a specified LDAP search key that are stored in the device's LDAP cache. The device sends an LDAP query to the LDAP server for the cached Attributes of the specified search key and replaces the old values in the cache with the new values received in the LDAP response.

For example, assume the cache contains a previously queried LDAP Attribute "telephoneNumber=1004" whose associated Attributes include "displayName", "mobile" and "ipPhone". If you perform a cache refresh based on the search key "telephoneNumber=1004", the device sends an LDAP query to the server requesting values for the "displayName", "mobile" and "ipPhone" Attributes of this search key. When the device receives the LDAP response, it replaces the old values in the cache with the new values received in the LDAP response.



➢ **To refresh the LDAP cache per LDAP Server Group:**

1. Open the LDAP Settings page (**Setup** menu > **IP Network** tab > **RADIUS & LDAP** folder > **LDAP Settings**).

2. Under the Cache Actions group, do the following:
   a. From the 'LDAP Group Index' drop-down list, select the required LDAP Server Group (see Configuring LDAP Server Groups).
   b. In the 'LDAP Refresh Cache by Key' field, enter the LDAP search key that you want to refresh (e.g., telephoneNumber=1004).
   c. Click **Refresh**; if a request with the specified key exists in the cache, a request is sent to the LDAP server for the Attributes associated in the cache with the search key.

Clearing the LDAP Cache

You can remove (clear) all LDAP entries in the device's LDAP cache for a specific LDAP Server Group, as described in the following procedure.

➢ **To clear the LDAP cache:**

1. Open the LDAP Settings page (**Setup** menu > **IP Network** tab > **RADIUS & LDAP** folder > **LDAP Settings**).
2. Under the Cache Actions group, do the following:
   a. From the 'LDAP Group Index' drop-down list, select the required LDAP Server Group (see Configuring LDAP Server Groups).
   b. Click **Clear Group**.

Configuring Local Database for Management User Authentication

You can configure the device to use the Local Users table (local database) to authenticate management users based on username-password combination. You can configure the device to use the Local Users table (see Configuring Management User Accounts) upon the following scenarios:

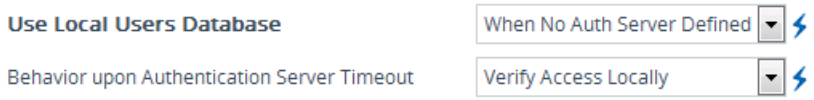
- LDAP or RADIUS server is not configured (or broken connection) or always use the Local Users table and only if the user is not found, to use the server.
- Connection with the LDAP or RADIUS server fails due to a timeout. In such a scenario, the device can deny access or verify the user's credentials (username-password) locally in the Local Users table.

If user authentication using the Local Users table succeeds, the device grants management access to the user; otherwise access is denied. The access level assigned to the user is also determined by the Local Users table.

- This feature is applicable to LDAP and RADIUS.
- This feature is applicable only to user management authentication.

➢ **To use the Local Users table for authenticating management users:**

1. Open the Authentication Server page (**Setup** menu > **Administration** tab > **Web & CLI** folder > **Authentication Server**).



2. Under the General group, do the following:
   a. Configure when the Local Users table must be used to authenticate login users. From the 'Use Local Users Database' drop-down list, select one of the following:
      - **When No Auth Server Defined (default):** When no LDAP/RADIUS server is configured or if a server is configured but connectivity with the server is down (if the server is up, the device authenticates the user with the server).

- **Always:** First attempts to authenticate the user using the Local Users table, but if not found, it authenticates the user with the LDAP/RADIUS server.

b. Configure whether the Local Users table must be used to authenticate login users upon connection timeout with the server. From the 'Behavior upon Authentication Server Timeout' drop-down list, select one of the following:

- **Deny Access:** User is denied access to the management platform.
- **Verify Access Locally (default):** The device verifies the user's credentials in the Local Users table.

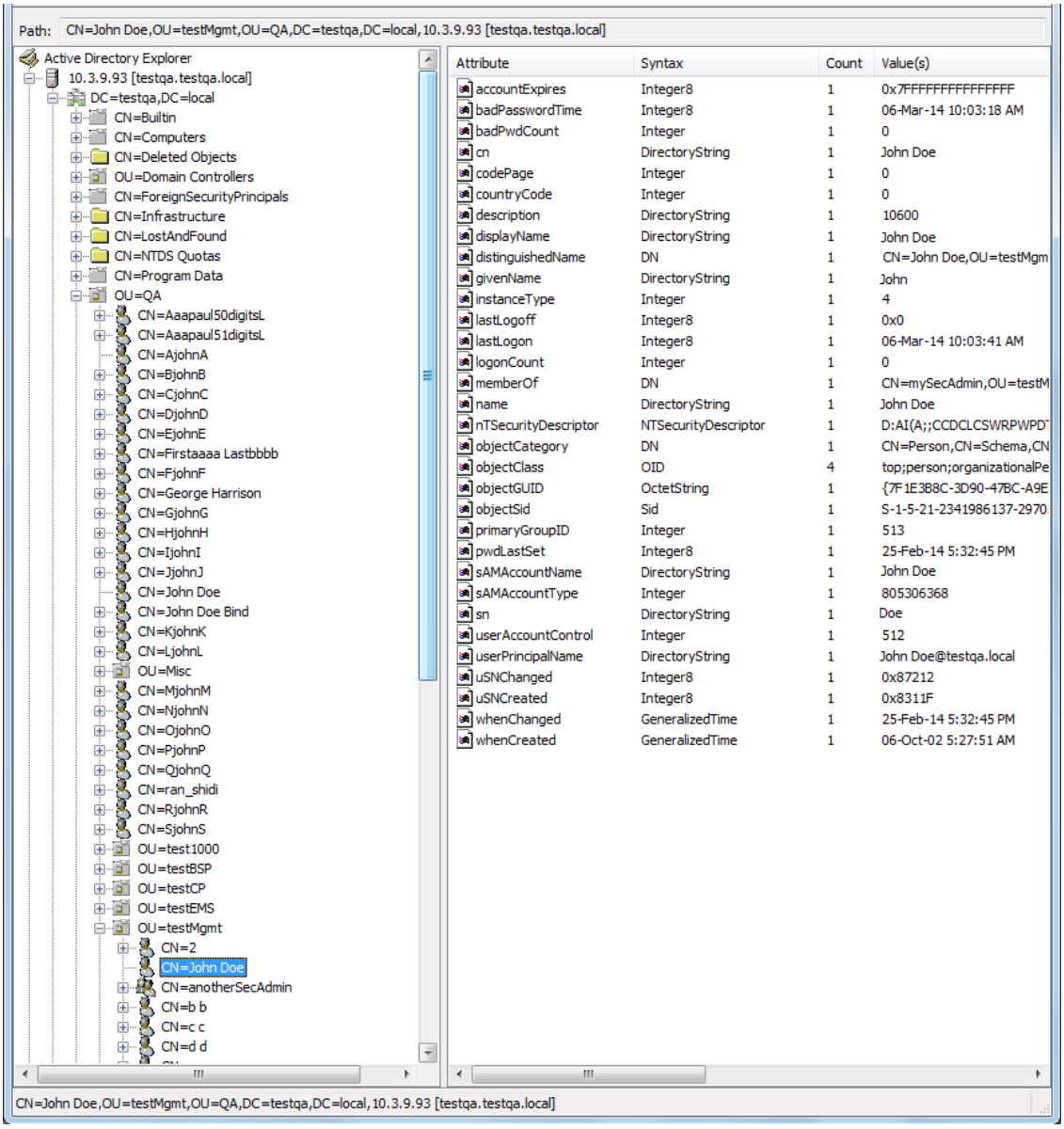
3. Click **Apply**, and then reset the device with a save-to-flash for your settings to take effect.

LDAP-based Login Authentication Example

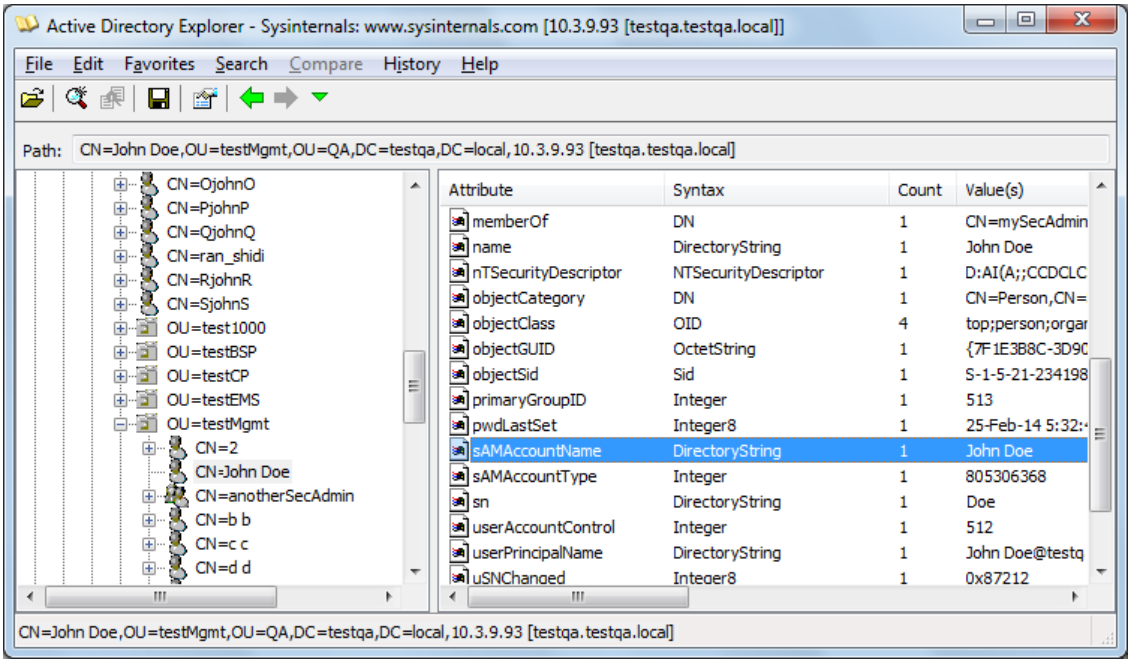
To facilitate your understanding on LDAP entry data structure and how to configure the device to use and obtain information from this LDAP directory, a brief configuration example is described in this section. The example applies to LDAP-based user login authentication and authorization (access level), and assumes that you are familiar with other aspects of LDAP configuration (e.g., LDAP server's address).

The LDAP server's entry data structure schema in the example is as follows:

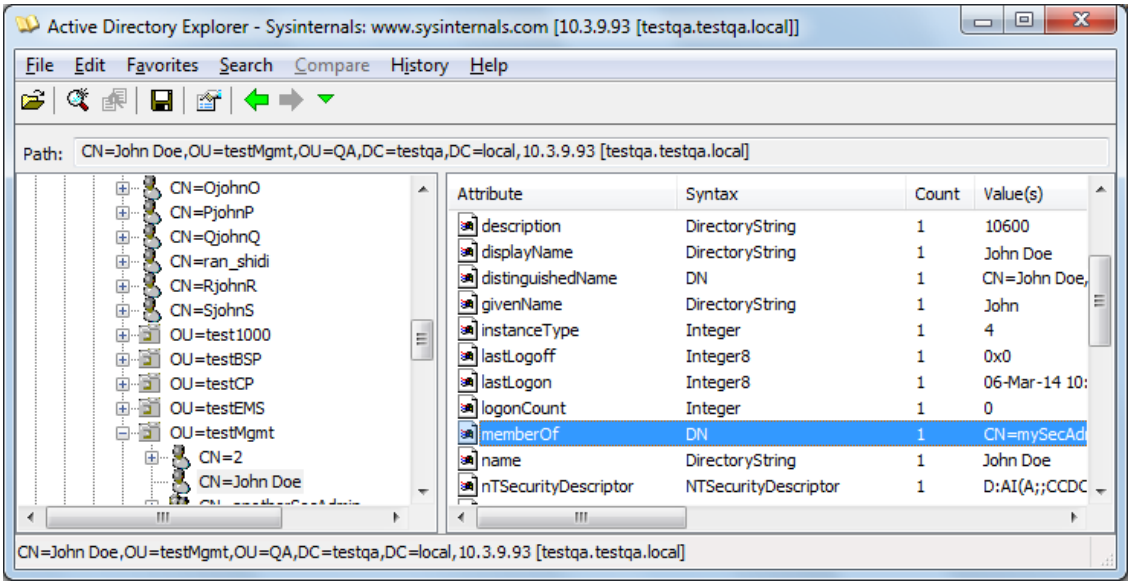
- **DN (base path):** OU=testMgmt,OU=QA,DC=testqa,DC=local. The DN path to search for the username in the directory is shown below:



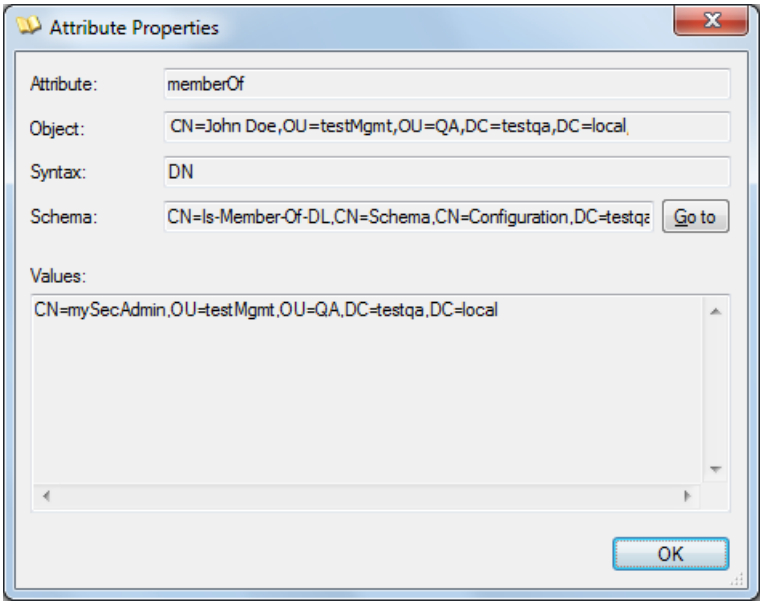
- **Search Attribute Filter:** (sAMAccountName=$). The login username is found based on this attribute (where the attribute's value equals the username):



- **Management Attribute:** memberOf. The attribute contains the member groups of the user:

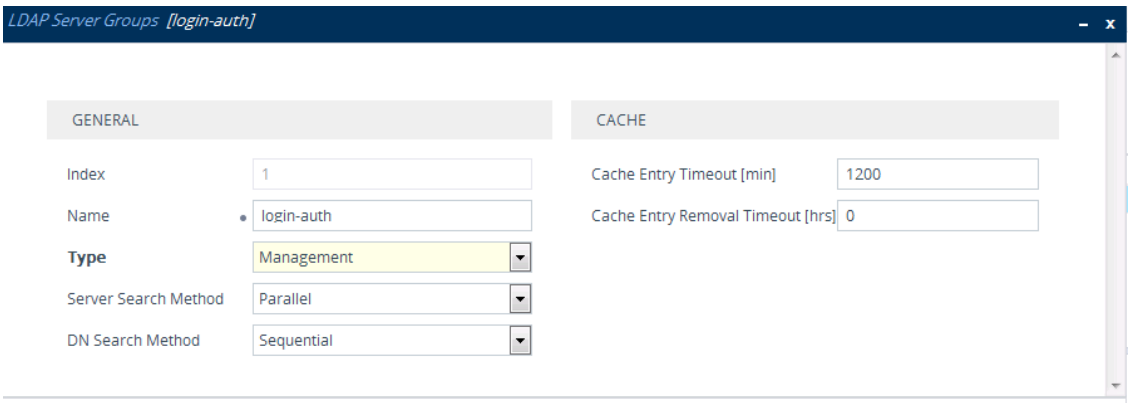


- **Management Group:** mySecAdmin. The group to which the user belongs, as listed under the memberOf attribute:

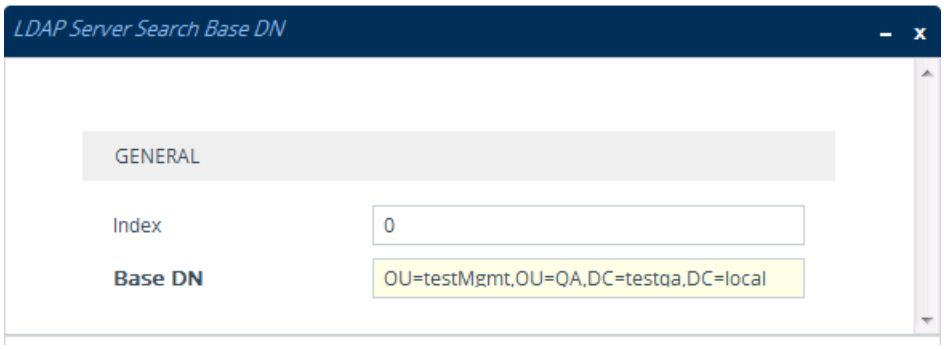


The configuration to match the above LDAP data structure schema is as follows:

- LDAP-based login authentication (management) is enabled in the LDAP Server Groups table (see Configuring LDAP Server Groups):

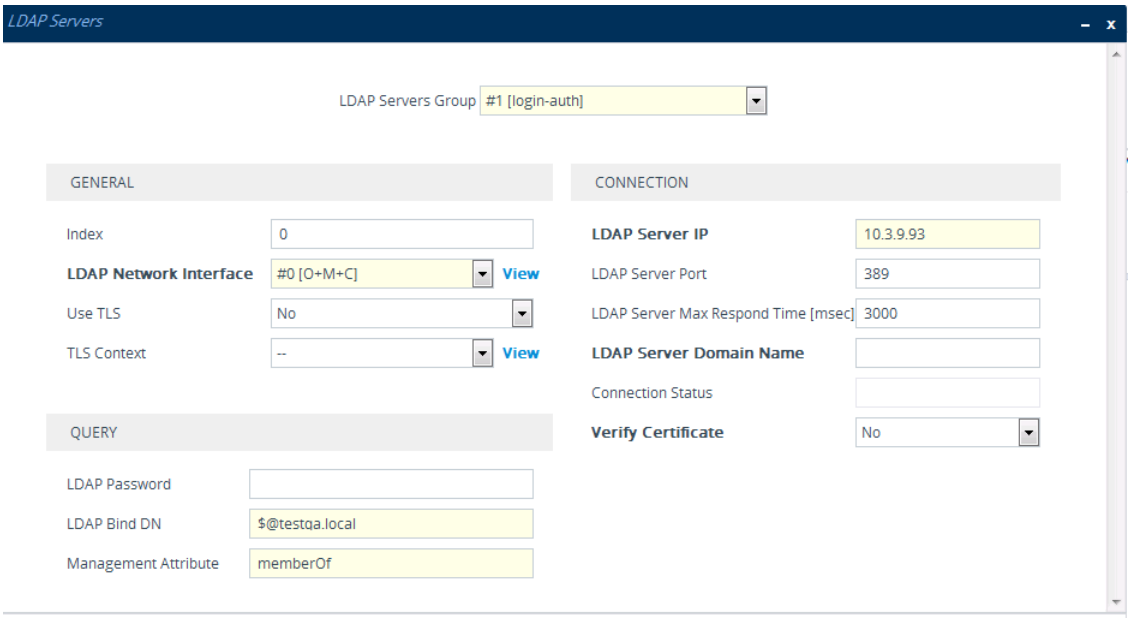


- The DN is configured in the LDAP Server Search Base DN table (see Configuring LDAP DNs (Base Paths) per LDAP Server):

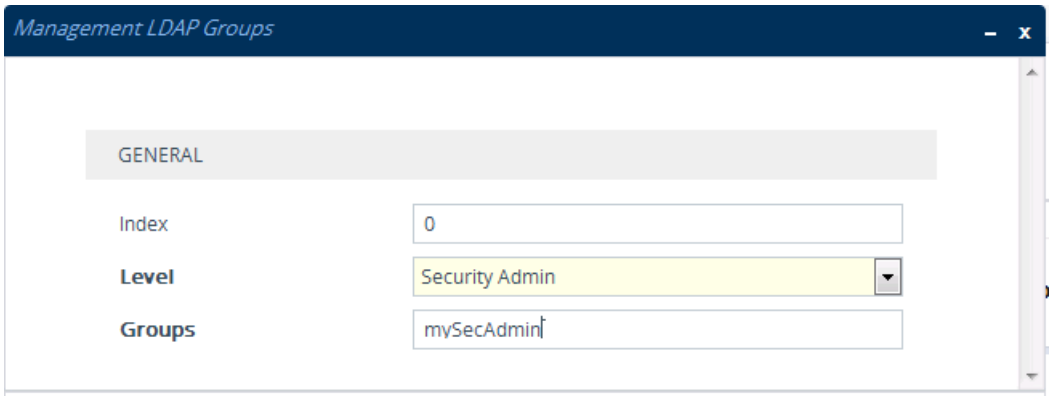


- The search attribute filter based on username is configured by the 'LDAP Authentication Filter' parameter (see Configuring the LDAP Search Filter Attribute):

- The group management attribute is configured by the 'Management Attribute' parameter in the LDAP Servers table:



- The management group and its corresponding access level is configured in the Management LDAP Groups table (see Configuring Access Level per Management Groups Attributes):



Enabling LDAP Searches for Numbers with Characters

Typically, the device performs LDAP searches in the AD for complete numbers where the digits are adjacent to one another (e.g., 5038234567). However, if the number is defined in the AD with characters (such as spaces, hyphens and periods) separating the digits (e.g., 503-823 4567), the LDAP query returns a failed result.

To enable the device to search the AD for numbers that may contain characters between its digits, you need to specify the Attribute (up to five) for which you want to apply this functionality, using the LDAPNumericAttributes parameter. For example, the telephoneNumber Attribute could be defined in AD with the telephone number "503-823-4567" (i.e., hyphens), "503.823.4567" (i.e., periods) or "503 823 4567" (i.e., spaces). If the device performs an LDAP search on this Attribute for the number 5038234567, the LDAP query will return results only if you configure the LDAPNumericAttributes parameter with the telephoneNumber Attribute (e.g., LDAPNumericAttributes=telephoneNumber). To search for the number with characters, the device inserts the asterisk (*) wildcard between all digits in the LDAP query (e.g., telephoneNumber = 5*0*3*8*2*3*4*5*6*7). As the AD server recognizes the * wildcard as representing any character, it returns all possible results to the device. Note that the wildcard represents only a character; a query result containing a digit in place of a wildcard is discarded and the device performs another query for the same Attribute. For example, it may return the numbers 533-823-4567 (second digit "3" and hyphens) and 503-823-4567. As the device discards query results where the wildcard results in a digit, it selects 503-823-4567 as the result. The correct query result is cached by the device for subsequent queries and/or in case of LDAP server failure.

AD-based Routing for Microsoft Teams or Skype for Business

Typically, companies wishing to deploy Microsoft® Teams or Skype for Business (formerly known as Lync) are faced with a complex, call routing dial plan when migrating users from their existing PBX or IP PBX to the Teams / Skype for Business platform. As more and more end-users migrate to the new voice system, dialing plan management and PBX link capacity can be adversely impacted. To resolve this issue, companies can employ Microsoft's Active Directory (AD), which provides a central database to manage and maintain information regarding user's availability, presence, and location.

The device supports outbound IP call routing decisions based on information stored on the AD. Based on queries sent to the AD, the device can route the call to one of the following IP domains:

- Teams / Skype for Business client - users connected to Teams / Skype for Business
- PBX or IP PBX - users not yet migrated to Teams / Skype for Business
- Mobile - mobile number
- Private - private telephone line for Teams / Skype for Business users (in addition to the primary telephone line)

This section describes an earlier implementation for configuring AD-based routing. For new deployments, it's **recommended** to use Call Setup Rules (see Configuring Call Setup Rules on page 595). Call Setup Rules provide more flexibility and easier implementation. You can view examples in Call Setup Rule Examples on page 605.

Querying the AD and Routing Priority

The device queries the AD using the initial destination number (i.e., called number). The query can return up to four user phone numbers, each pertaining to one of the IP domains (i.e., private number, Skype for Business number, PBX / IP PBX number, and mobile number). The configuration parameters listed in the table below are used to configure the query attribute keys that defines the AD attribute that you wish to query in the AD:

Table 16-13:Parameters for Configuring Query Attribute Key

Parameter	Queried User Domain (Attribute) in AD	Query or Query Result Example
MSLDAPPBXNumAttributeName	PBX or IP PBX number (e.g., "telephoneNumber" - default)	telephoneNumber= +3233554447
MSLDAPOCSNumAttributeName	Mediation Server / Skype for Business client number (e.g., "msRTCSIP-Line")	msRTCSIP-Line=john.smith@company.com
MSLDAPMobileNumAttributeName	Mobile number (e.g., "mobile")	mobile=+3247647156
MSLDAPPrivateNumAttributeName	Any attribute (e.g., "msRTCSIP-PrivateLine") **Note:** Used only if set to same value as Primary or Secondary key.	msRTCSIP-PrivateLine= +3233554480
MSLDAPPrimaryKey	Primary Key query search instead of PBX key - can be any AD attribute	msRTCSIP-PrivateLine= +3233554480
MSLDAPSecondaryKey	Secondary Key query key search if Primary Key fails - can be any attribute	-

The process for querying the AD and subsequent routing based on the query results is as follows:

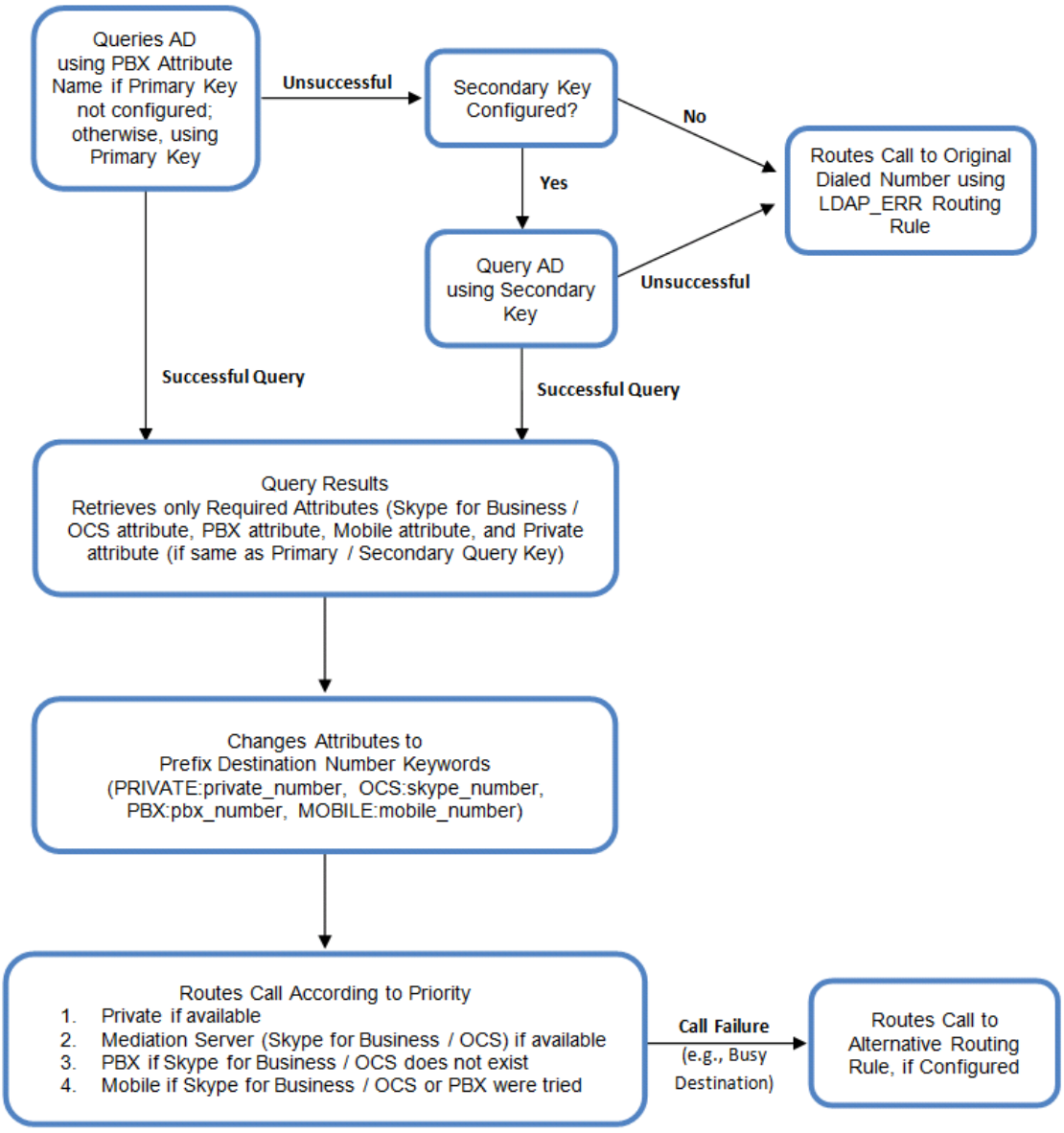
1. If the Primary Key is configured, it uses the defined string as a primary key instead of the one defined in MSLDAPPBXNumAttributeName. It requests the attributes which are described below.
2. If the primary query is not found in the AD and the Secondary Key is configured, it does a second query for the destination number using a second AD attribute key name, configured by the MSLDAPSecondaryKey parameter.
3. If none of the queries are successful, it routes the call to the original dialed destination number according to the routing rule matching the "LDAP_ERR" destination prefix number value, or rejects the call with a SIP 404 "Not Found" response.
4. For each query (primary or secondary), it queries the following attributes (if configured):
   - MSLDAPPBXNumAttributeName
   - MSLDAPOCSNumAttributeName
   - MSLDAPMobileNumAttributeName

   In addition, it queries the special attribute defined in MSLDAPPrivateNumAttributeName, only if the query key (primary or secondary) is equal to its value.
5. If the query is found: The AD returns up to four attributes - Skype for Business, PBX / IP PBX, private (only if it equals Primary or Secondary key), and mobile.
6. The device adds unique prefix keywords to the query results in order to identify the query type (i.e., IP domain). These prefixes are used as the prefix destination number value in the Tel-to-IP Routing table to denote the IP domains:
   - "PRIVATE" (PRIVATE:<private_number>): used to match a routing rule based on query results of the private number (MSLDAPPrivateNumAttributeName)
   - "OCS" (OCS:<Skype for Business_number>): used to match a routing rule based on query results of the Skype for Business client number (MSLDAPOCSNumAttributeName)
   - "PBX" (PBX:<PBX_number>): used to match a routing rule based on query results of the PBX / IP PBX number (MSLDAPPBXNumAttributeName)
   - "MOBILE" (MOBILE:<mobile_number>): used to match a routing rule based on query results of the mobile number (MSLDAPMobileNumAttributeName)
   - "LDAP_ERR": used to match a routing rule based on a failed query result when no attribute is found in the AD

These prefixes are involved only in the routing and manipulation processes; they are not used as the final destination number.

7. The device uses the Tel-to-IP Routing table to route the call based on the LDAP query result. The device routes the call according to the following priority:
   a. **Private line:** If the query is done for the private attribute and it's found, the device routes the call according to this attribute.
   b. **Mediation Server SIP address (Skype for Business):** If the private attribute does not exist or is not queried, the device routes the call to the Mediation Server (which then routes the call to the Skype for Business client).
   c. **PBX / IP PBX:** If the Skype for Business client is not found in the AD, it routes the call to the PBX / IP PBX.
   d. **Mobile number:** If the Skype for Business client (or Mediation Server) is unavailable (e.g., SIP response 404 "Not Found" upon INVITE sent to Skype for Business client), and the PBX / IP PBX is also unavailable, the device routes the call to the user's mobile number (if exists in the AD).
   e. **Alternative route:** If the call routing to all the above fails (e.g., due to unavailable destination - call busy), the device can route the call to an alternative destination if an alternative routing rule is configured.
   f. **"Redundant" route:** If the query failed (i.e., no attribute found in the AD), the device uses the routing rule matching the "LDAP_ERR" prefix destination number value.

The flowchart below summarizes the device's process for querying the AD and routing the call based on the query results:



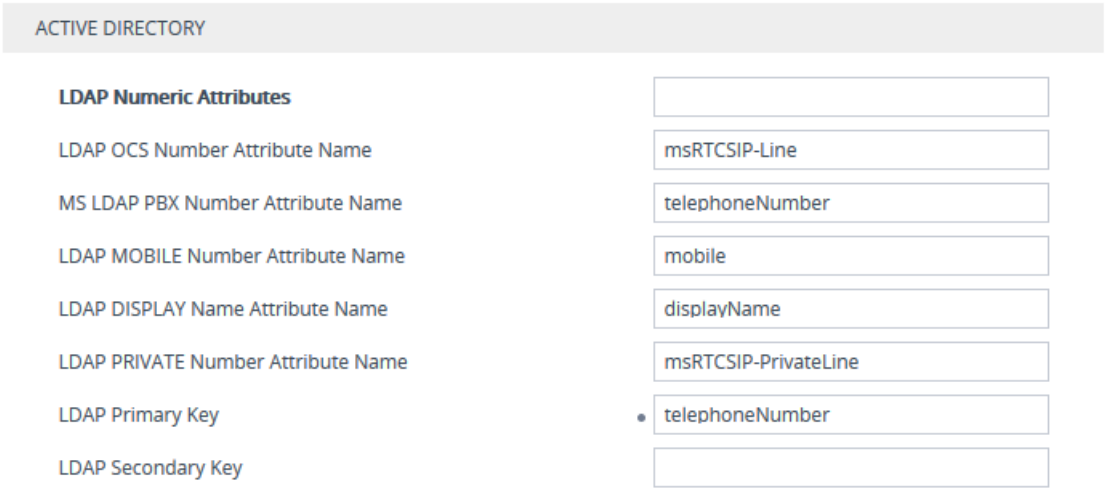
If you are using the device's local LDAP cache, see Configuring the Device's LDAP Cache for the LDAP query process.

Configuring AD-Based Routing Rules

The following procedure describes how to configure outbound IP routing based on LDAP queries.

➢ **To configure LDAP-based IP routing for Skype for Business:**

1. Configure the LDAP server parameters, as described in Configuring LDAP Servers.
2. Configure the AD attribute names used in the LDAP query:
   a. Open the LDAP Settings page (**Setup** menu > **IP Network** tab > **RADIUS & LDAP** folder > **LDAP Settings**).



b. Configure the LDAP attribute names as desired.

3. SBC application: Configure AD-based IP-to-IP routing rules:

   a. Open the IP-to-IP Routing table (see Configuring SBC IP-to-IP Routing Rules).

   b. Configure query-result routing rules for each IP domain (private, PBX / IP PBX, Skype for Business clients, and mobile), using the LDAP keywords (case-sensitive) in the 'Destination Username Pattern' field:

      - PRIVATE: Private number
      - OCS: Skype for Business client number
      - PBX: PBX / IP PBX number
      - MOBILE: Mobile number
      - LDAP_ERR: LDAP query failure

   c. Configure a routing rule for routing the initial call (LDAP query) to the LDAP server, by setting the 'Destination Type' field to LDAP for denoting the IP address of the LDAP server.

   d. For alternative routing, enable the alternative routing mechanism and configure corresponding SIP reasons for alternative routing. For this feature, alternative routing starts from the table row located under the LDAP query row.

The table below shows an example for configuring AD-based SBC routing rules in the IP-to-IP Routing Table:

Table 16-14:AD-Based SBC IP-to-IP Routing Rule Configuration Examples

Index	Destination Username Pattern	Destination Type	Destination Address
1	PRIVATE:	Dest Address	10.33.45.60
2	PBX:	Dest Address	10.33.45.65
3	OCS:	Dest Address	10.33.45.68

Index	Destination Username Pattern	Destination Type	Destination Address
4	MOBILE:	Dest Address	10.33.45.100
5	LDAP_ERR	Dest Address	10.33.45.80
6	*	LDAP	-
7	*	Dest Address	10.33.45.72

The configured routing rule example is explained below:

- **Rule 1:** Sends call to private telephone line (at 10.33.45.60) upon successful AD query result for the private attribute.
- **Rule 2:** Sends call to IP PBX (at 10.33.45.65) upon successful AD query result for the PBX attribute.
- **Rule 3:** Sends call to Skype for Business client (i.e., Mediation Server at 10.33.45.68) upon successful AD query result for the Skype for Business attribute.
- **Rule 4:** Sends call to user's mobile phone number (to PSTN through the device's IP address at 10.33.45.100) upon successful AD query result for the Mobile attribute.
- **Rule 5:** Sends call to IP address of device (10.33.45.80) if AD query failure (e.g., no response from LDAP server or attribute not found).
- **Rule 6:** Sends query for original destination number of received call to the LDAP server.
- **Rule 7:** Alternative routing rule that sends the call of original dialed number to IP destination 10.33.45.72. This rule is applied in any of the following cases
  - LDAP functionality is disabled.
  - LDAP query is successful but call fails (due to, for example, busy line) to all the relevant attribute destinations (private, Skype for Business, PBX, and mobile), and a relevant SBC Alternative Routing Reason (see Configuring SIP Response Codes for Alternative Routing Reasons) has been configured.

Once the device receives the original incoming call, the first rule that it uses is Rule 6, which queries the AD server. When the AD replies, the device searches the table, from the first rule down, for the matching destination phone prefix (i.e., "PRIVATE:, "PBX:", "OCS:", "MOBILE:", and "LDAP_ERR:"), and then sends the call to the appropriate destination.

Least Cost Routing

This section describes the device's Least Cost Routing (LCR) feature.

Overview

The LCR feature enables the device to choose the outbound IP destination routing rule based on lowest call cost. This is useful in that it enables service providers to optimize routing costs

for customers. For example, you may wish to define different call costs for local and international calls or different call costs for weekends and weekdays (specifying even the time of call). The device sends the calculated cost of the call to a Syslog server (as Information messages), thereby enabling billing by third-party vendors.

LCR is implemented by defining Cost Groups and assigning them to routing rules in the IP-to-IP Routing table (for SBC calls). The device searches the routing table for matching routing rules and then selects the rule with the lowest call cost. If two routing rules have identical costs, the rule appearing higher up in the table is used (i.e., first-matched rule). If the selected route is unavailable, the device selects the next least-cost routing rule.

Even if a matched routing rule is not assigned a Cost Group, the device can select it as the preferred route over other matched rules that are assigned Cost Groups. This is determined according to the settings of the 'Default Call Cost' parameter configured for the Routing Policy (associated with the routing rule for SBC calls). To configure the Routing Policy,; for SBC calls, see Configuring SBC Routing Policy Rules.

The Cost Group defines a fixed connection cost (*connection cost*) and a charge per minute (*minute cost*). Cost Groups can also be configured with time segments (*time bands*), which define connection cost and minute cost based on specific days of the week and time of day (e.g., from Saturday through Sunday between 6:00 and 18:00, and Monday through Sunday between 18:00 and 5:00). If multiple time bands are configured per Cost Group and a call spans multiple time bands, the call cost is calculated according to minute cost per time band and the connection cost of the time band in which the call was initially established.

In addition to Cost Groups, the device can calculate the call cost using an optional, user-defined average call duration value. The logic in using this option is that a Cost Group may be cheap if the call duration is short, but due to its high minute cost, may prove very expensive if the duration is lengthy. Thus, together with Cost Groups, the device can use this option to determine least cost routing. The device calculates the Cost Group call cost as follows:

*Total Call Cost = Connection Cost + (Minute Cost * Average Call Duration)*

The below table shows an example of call cost when taking into consideration call duration. This example shows four defined Cost Groups and the total call cost if the average call duration is 10 minutes:

Table 16-15:Call Cost Comparison between Cost Groups for different Call Durations

Cost Group	Connection Cost	Minute Cost	Total Call Cost per Duration	
			1 Minute	10 Minutes
A	1	6	7	61
B	0	10	10	100
C	0.3	8	8.3	80.3
D	6	1	7	16

If four matching routing rules are located in the routing table and each one is assigned a different Cost Group as listed in the table above, then the rule assigned Cost Group "D" is selected. Note that for one minute, Cost Groups "A" and "D" are identical, but due to the average call duration, Cost Group "D" is cheaper. Therefore, average call duration is an important factor in determining the cheapest routing role.

Below are a few examples of how you can implement LCR:

- **Example 1:** This example uses two different Cost Groups for routing local calls and international calls:

  Two Cost Groups are configured as shown below:

Cost Group	Connection Cost	Minute Cost
1. "Local Calls"	2	1
2. "International Calls"	6	3

  The Cost Groups are assigned to routing rules for local and international calls:

Routing Index	Dest Phone Prefix	Destination IP	Cost Group ID
1	2000	x.x.x.x	1 "Local Calls"
2	00	x.x.x.x	2 "International Calls"

- **Example 2:** This example shows how the device determines the cheapest routing rule in the Tel-to-IP Routing table:

  The 'Default Call Cost' parameter in the Routing Policy rule is configured to **Lowest Cost**, meaning that if the device locates other matching routing rules (with Cost Groups assigned), the routing rule without a Cost Group is considered the lowest cost route.

  - The following Cost Groups are configured:

Cost Group	Connection Cost	Minute Cost
1. "A"	2	1
2. "B"	6	3

  - The Cost Groups are assigned to routing rules:

Routing Index	Dest Phone Prefix	Destination IP	Cost Group
1	201	x.x.x.x	"A'
2	201	x.x.x.x	"B"

Routing Index	Dest Phone Prefix	Destination IP	Cost Group
3	201	x.x.x.x	0
4	201	x.x.x.x	"B"

The device calculates the optimal route in the following index order: 3, 1, 2, and then 4, due to the following logic:

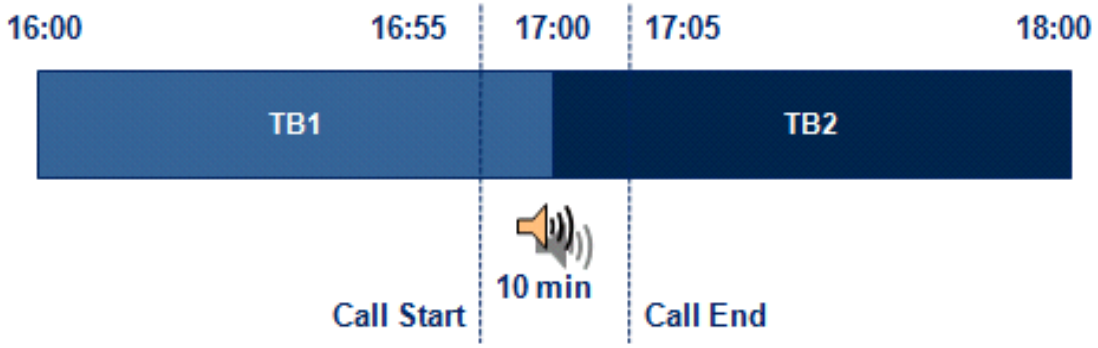
- Index 1 - Cost Group "A" has the lowest connection cost and minute cost
- Index 2 - Cost Group "B" takes precedence over Index 4 entry based on the first-matched method rule
- Index 3 - no Cost Group is assigned, but as the 'Default Call Cost' parameter is configured to **Lowest Cost**, it is selected as the cheapest route
- Index 4 - Cost Group "B" is only second-matched rule (Index 1 is the first)

Example 3: This example shows how the cost of a call is calculated if the call spans over multiple time bands:

Assume a Cost Group, "CG Local" is configured with two time bands, as shown below:

Cost Group	Time Band	Start Time	End Time	Connection Cost	Minute Cost
CG Local	TB1	16:00	17:00	2	1
TB2	17:00	18:00	7	2	

Assume that the call duration is 10 minutes, occurring between 16:55 and 17:05. In other words, the first 5 minutes occurs in time band "TB1" and the next 5 minutes occurs in "TB2", as shown below:



The device calculates the call cost as follows:

- For the first 5 minutes of the call (16:55 to 17:00), the call is in time band "TB1" and the call cost for this period is calculated as follows:

  Connection Cost of "TB1" + [Minute Cost of "TB1" x call duration] = 2 + [1 x 5 min] = ***7***

- For the next 5 minutes of the call (17:00 to 17:05), the call is in time band "TB2" and the call cost for this period is calculated as follows:

Minute Cost of "TB2" x call duration = 2 x 5 min = ***10***

- Therefore, the total call cost is the summation of above:

  "TB1" call cost + "TB2" call cost = 7 + 10 = ***17***

Configuring LCR

To configure LCR, perform the following main steps:

1. Enable LCR:
   - SBC application: Configuring SBC Routing Policy Rules
2. Configure Cost Groups - see Configuring Cost Groups.
3. Configure Time Bands for a Cost Group - see Configuring Time Bands for Cost Groups.
4. Assign Cost Groups to outbound IP routing rules - see Assigning Cost Groups to Routing Rules.

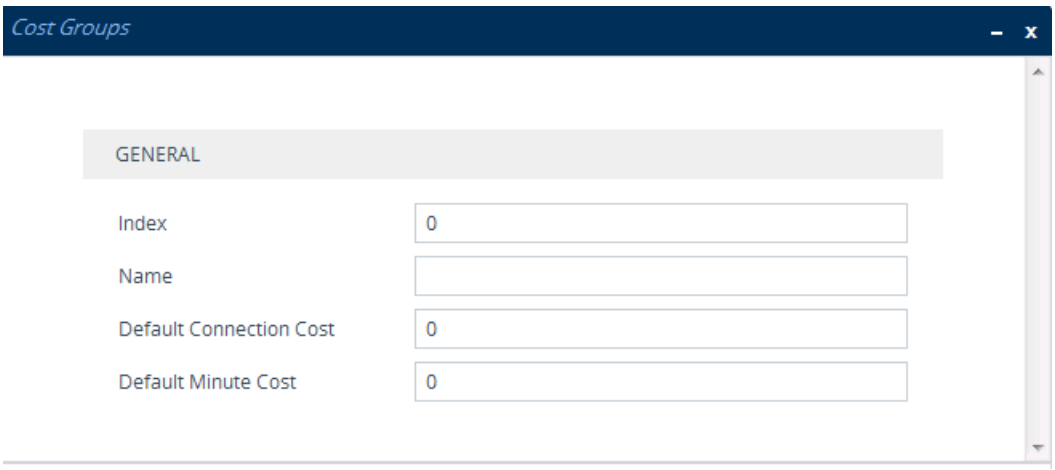
Configuring Cost Groups

The Cost Groups table lets you configure up to 10 Cost Groups. A Cost Group defines a fixed call connection cost and a call rate (charge per minute). Once configured, you can configure Time Bands per Cost Group.

The following procedure describes how to configure Cost Groups through the Web interface. You can also configure it through ini file [CostGroupTable] or CLI (`configure voip > sip-definition least-cost-routing cost-group`).

➢ **To configure a Cost Group:**

1. Open the Cost Groups table (**Setup** menu > **Signaling & Media** tab > **SIP Definitions** folder > **Least Cost Routing** > **Cost Groups**).
2. Click **New**; the following dialog box appears:



3. Configure a Cost Group according to the parameters described in the table below.
4. Click **Apply**, and then save your settings to flash memory.

Table 16-16:Cost Groups Table Parameter Descriptions

Parameter	Description
'Index' [CostGroupTable_Index]	Defines an index number for the new table row. **Note:** Each row must be configured with a unique index.
'Name' `cost-group-name` [CostGroupTable_ CostGroupName]	Defines a descriptive name, which is used when associating the row in other tables. The valid value is a string of up to 40 characters. **Note:** ■ Each row must have a unique name. ■ The parameter value cannot contain a forward slash (/).
'Default Connection Cost' `default-connection-cost` [CostGroupTable_ DefaultConnectionCost]	Defines the call connection cost (added as a fixed charge to the call) for a call outside the time bands. The valid value range is 0-65533. The default is 0. **Note:** When calculating the cost of a call, if the current time of the call is not within a time band configured for the Cost Group, then this default connection cost is used.
'Default Minute Cost' `default-minute-cost` [CostGroupTable_ DefaultMinuteCost]	Defines the call charge per minute for a call outside the time bands. The valid value range is 0-65533. The default is 0. **Note:** When calculating the cost of a call, if the current time of the call is not within a time band configured for the Cost Group, then this default charge per minute is used.

Configuring Time Bands for Cost Groups

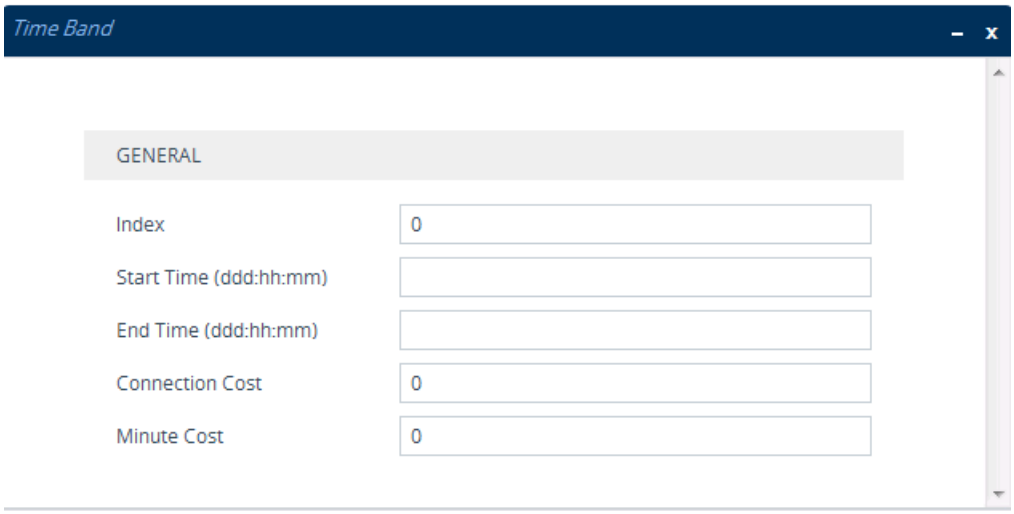
The Time Band table lets you configure Time Bands per Cost Group. A Time Band defines a day and time range (e.g., from Saturday 05:00 to Sunday 24:00) and a fixed call connection charge and call rate per minute for this interval. You can configure up to 70 Time Bands, where up to 21 Time Bands can be assigned to each Cost Group.

- You cannot configure overlapping Time Bands.
- If a Time Band is not configured for a specific day and time range, the default connection cost and default minute cost configured for the Cost Group in the Cost Groups table is applied.

The following procedure describes how to configure Time Bands per Cost Group through the Web interface. You can also configure it through ini file [CostGroupTimebands] or CLI (`configure voip > sip-definition least-cost-routing cost-group-time-bands`).

➢ **To configure a Time Band per Cost Group:**

1. Open the Cost Groups table (**Setup** menu > **Signaling & Media** tab > **SIP Definitions** folder > **Least Cost Routing** > **Cost Groups**).
2. Select a Cost Group for which you want to assign Time Bands, and then click the **Time Band** link located below the table; the Time Band table for the selected Cost Group appears.
3. Click **New**; the following dialog box appears:



4. Configure a Time Band according to the parameters described in the table below.
5. Click **Apply**, and then save your settings to flash memory.

Table 16-17:Time Band Table Description

Parameter	Description
'Index' `timeband-index` [CostGroupTimebands_ TimebandIndex]	Defines an index number for the new table row. **Note:** Each row must be configured with a unique index.
'Start Time' `start-time` [CostGroupTimebands_ StartTime]	Defines the day and time of day from when this time band is applicable. The format is DDD:hh:mm, where: ■ *DDD* is the day of the week, represented by the first three letters of the day in uppercase (i.e., SUN, MON, TUE, WED, THU, FRI, or SAT). ■ *hh* and *mm* denote the time of day, where *hh* is the hour (00-23) and *mm* the minutes (00-59) For example, SAT:22:00 denotes Saturday at 10 pm.
'End Time'	Defines the day and time of day until when this time band is

Parameter	Description
`end-time` [CostGroupTimebands_EndTime]	applicable. For a description of the valid values, see the parameter above.
'Connection Cost' `connection-cost` [CostGroupTimebands_ConnectionCost]	Defines the call connection cost during the time band. This is added as a fixed charge to the call. The valid value range is 0-65533. The default is 0. **Note:** The entered value must be a whole number (i.e., not a decimal).
'Minute Cost' `minute-cost` [CostGroupTimebands_MinuteCost]	Defines the call cost per minute charge during the time band. The valid value range is 0-65533. The default is 0. **Note:** The entered value must be a whole number (i.e., not a decimal).

Assigning Cost Groups to Routing Rules

To use your configured Cost Groups, you need to assign them to routing rules:

- SBC application: IP-to-IP Routing table - see Configuring SBC IP-to-IP Routing on page 716

Remote Web Services

This section describes configuration for remote Web services.

To debug remote Web services, see Debugging Web Services.

Configuring Remote Web Services

The Remote Web Services table lets you configure up to seven Web-based (HTTP/S) services (*Remote Web Services*) provided by third-party, remote HTTP/S hosts (*HTTP Remote Hosts*). The following types of services can be offered by the remote hosts: Routing service, Call Status service, Topology Status service, QoS service, General service, and Registration Status service. For more information on these services, see the description of the 'Type' parameter below.

A Remote Web Service is configured using two tables with "parent-child" relationship:

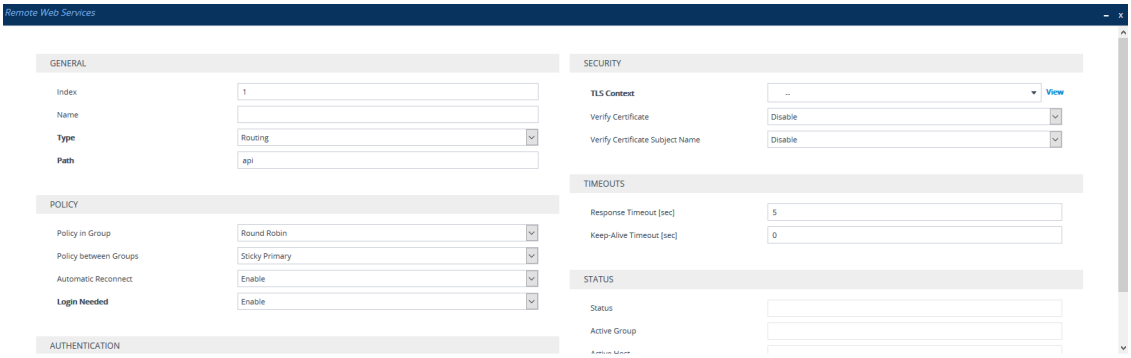
- Remote Web Services table ("parent"): Defines the name of the Remote Web Service as well as other settings (e.g., type of service). This table is described below.
- HTTP Remote Hosts table ("child"): Defines remote HTTP hosts (e.g., IP address) per Remote Web Service. For more information, see Configuring Remote HTTP Hosts on page 325.

- You can configure only **one** Remote Web Service for each of the following service types: **Routing**, **Call Status**, **Topology Status**, **QoS**, **Registration Status**, and **Remote Monitoring**.
- The Routing service also includes the Call Status and Topology Status services.
- The device supports HTTP redirect responses (3xx) only during connection establishment with the host. Upon receipt of a redirect response, the device attempts to open a new socket with the host and if this is successful, closes the current connection.

The following procedure describes how to configure Remote Web Services through the Web interface. You can also configure it through ini file [HTTPRemoteServices] or CLI (`configure system > http-services > http-remote-services`).

➢ To configure a remote Web service:

1. Open the Remote Web Services table (**Setup** menu > **IP Network** tab > **Web Services** folder > **Remote Web Services**).
2. Click **New**; the following dialog box appears:



3. Configure a remote Web service according to the parameters described in the table below.
4. Click **Apply**, and then save your settings to flash memory.

Table 16-18:Remote Web Services Table Parameter Descriptions

Parameter	Description
General	
'Index' [HTTPRemoteServices_Index]	Defines an index number for the new table row. **Note:** ■ Each row must be configured with a unique index. ■ The parameter is mandatory.
'Name' `rest-name`	Defines a descriptive name, which is used when associating the row in other tables.

Parameter	Description
[HTTPRemoteServices_Name]	The valid value is a string of up to 40 characters. **Note:** ■ Each row must be configured with a unique name. ■ The parameter is mandatory. ■ The parameter value cannot contain a forward slash (/).
'Type' `rest-message-type` [HTTPRemoteServices_ HTTPType]	Defines the type of service provided by the HTTP remote host: ■ [0] **Routing** = (Default) This option provides a call routing service, whereby the host (e.g., Routing server) determines the next hop of an incoming call on the path to its final destination. For more information on employing a third-party, routing server, see Centralized Third-Party Routing Server. This option also includes the services provided by the **Call Status** and **Topology Status** options. ■ [1] **Call Status** = This option provides a call status service for calls processed by the device. The device provides call status to the host by sending CDRs. ■ [2] **Topology Status** = This option provides a topology status service, which refers to all device configuration changes (add, edit and delete actions). The device sends topology status to the HTTP host, using the REST API command, TopologyStatus. For this service to be functional, you also need to enable the Topology Status service as described in Enabling Topology Status Services. Topology status includes the following: ✓ IP Group Connectivity: Status is reported when the keep-alive mechanism, enabled for the associated Proxy Set, detects that the IP Group is unavailable, or when CAC thresholds (configured in the Admission Control table) associated with the IP Group are crossed. ✓ Configuration Status: Status is reported when

Parameter	Description
	IP Groups or SIP Interfaces that are configured to be used by remote Web-based services (i.e., the UsedByRoutingServer parameter is set to 1 - Used) are created or deleted. If you subsequently change the settings of the UsedByRoutingServer parameter or the 'Name' parameter, the device reports the change as a creation or deletion of the corresponding configuration entity. ■ [5] **QoS** = This option provides a call routing service based on Quality of Service (QoS). For more information, see Configuring QoS-Based Routing by Routing Server. ■ [8] **General** = This option can be used for the following services: ✓ Generating and sending CDRs to a REST server through REST API. The REST server is configured as an HTTP-based server (Remote Web Service). For more information, see Configuring CDR Reporting to REST Server on page 986. ✓ Querying (GET) HTTP servers using Call Setup Rules. The response from the server can be used for various functionality such as tag-based classification and routing. When configuring the Call Setup Rule, you need to configure the 'Request Target' parameter to the name of this Remote Web Service. For more information on Call Setup Rules, see Configuring Call Setup Rules on page 595. ✓ Requesting a Push Notification Server to wake a SIP user agent (typically, a mobile device) that is registered with the server for Push Notification Service, through REST API. The REST server (Push Notification Server) is configured as an HTTP-based server (Remote Web Service). For more information, see Configuring Push Notification Service on page 812. ■ [9] **Registration Status** = This option provides a call

Parameter	Description
	routing service based on registration status. The device periodically synchronizes its database of registered user agents (endpoints) with the third-party Routing server (HTTP host) to keep it up to date, enabling the Routing server to use this information to perform correct and optimal routing decisions. For this service to be functional, you also need to enable the Registration Status service as described in Enabling Registration Status Services on page 328. ■ [10] **Remote Monitoring** = This option provides a remote monitoring of the device service when the device is located behind a NAT. The device sends its monitoring reports to this Remote Web Service (HTTP host). To enable remote monitoring and to select the report types that you want sent, see Remote Monitoring of Device behind NAT on page 1056. **Note:** ■ You can configure only one Remote Web Service for each of the following service types: **Routing**, **Call Status**, **Topology Status**, **QoS**, **Registration Status**, and **Remote Monitoring**. ■ The **Routing** option also includes the Call Status and Topology Status services. ■ If you don't configure the parameter to **QoS**, the device sends QoS reports to the Topology server. ■ For the Registration Status service, if you have not configured the parameter to **Registration Status** for any Remote Web Service, the device provides the service to the Remote Web Service for which you have configured the parameter to **Topology Status**.
'Path' `rest-path` [HTTPRemoteServices_Path]	Defines the path (prefix) to the REST APIs. The valid value is a string of up to 80 characters. The default is "api".
Policy	

Parameter	Description
'Policy in Group' `http-policy` [HTTPRemoteServices_Policy]	Defines the mode of operation between hosts in a group, which are configured in the HTTP Remote Hosts table for the specific remote Web service. ■ [0] **Round Robin** = (Default) The device does load balancing of traffic across all the hosts in the group. Every consecutive message is sent to the next available host. The priority of the hosts determines the order in which the device sends the traffic. ■ [1] **Sticky Primary** = The device always attempts to send traffic to the host that has the highest priority in the group. If the host does not respond, the device sends the traffic to the next available host that has the highest priority. If the host that has the highest priority becomes available again, the device sends the traffic to this host. ■ [2] **Sticky Next** = The device initially attempts to send traffic to the host that has the highest priority in the group. If this host becomes unavailable (or is initially unavailable), the device sends the traffic to the next available host that has the highest priority and continues sending traffic to this host even if the highest-priority host later becomes available again. **Note:** If you have configured multiple hosts with the same priority, their priority is determined by their order of appearance in the HTTP Remote Hosts table. For example, if two hosts are configured in rows Index 0 and Index 1 with priority 0, the host in Index 0 is considered higher priority.
'Policy between Groups' `http-policy-between-groups` [HTTPRemoteServices_ BetweenGroupsPolicy]	Defines the mode of operation between groups of hosts, which are configured in the HTTP Remote Hosts table for the specific remote Web service. ■ [1] **Sticky Primary** = (Default) The device always attempts to send traffic to the group that has the highest priority (e.g., Group 0). If none of the hosts in this group respond, the device attempts to send traffic to a host in a group that has the next highest priority (e.g., Group 1), and so on. Whenever a host in the group that has the highest priority (e.g., Group 0) becomes available again, the device sends

Parameter	Description
	the traffic to the host in this group. ■ [2] **Sticky Next** = The device initially attempts to send traffic to the group of hosts that has the highest priority (e.g., Group 0). If none of the hosts in the group respond, the device attempts to send traffic to a host in a group that has the next highest priority (e.g., Group 1). Even if the group of hosts that has the highest priority (e.g., Group 0) becomes available again, the device continues sending traffic to this lower priority group (e.g., Group 1) .
'Automatic Reconnect' `http-persistent-connection` [HTTPRemoteServices_ PersistentConnection]	Defines whether the HTTP connection with the host remains open or is only opened per request. ■ [0] **Disable** = The HTTP connection is created per client (user) request and remains connected until the server closes the connection. ■ [1] **Enable** = (Default) The device creates the HTTP connection once you have configured the service. If the server closes the connection, the device re-opens it. If the keep-alive timeout is configured, the device uses HTTP keep-alive messages to keep the connection open all the time.
Login Needed `http-login-needed` [HTTPRemoteServices_ LoginNeeded]	Enables the use of the AudioCodes proprietary REST API Login and Logout commands for connecting to the remote host. The commands verify specific information (e.g., software version) before allowing connectivity with the device. ■ [0] **Disable** = Commands are not used. ■ [1] **Enable** (default) **Note:** The parameter is applicable only if you configure the 'Type' parameter to any value other than **General**.
Authentication	
'Username' `rest-user-name` [HTTPRemoteServices_ AuthUserName]	Defines the username for HTTP authentication. The valid value is a string of up to 80 characters. The default is "user".

Parameter	Description
'Password' rest-password [HTTPRemoteServices_ AuthPassword]	Defines the password for HTTP authentication. The valid value is a string of up to 80 characters. The default is "password". **Note:** The password cannot be configured with wide characters.
Security	
'TLS Context' rest-tls-context [HTTPRemoteServices_ TLSContext]	Assigns a TLS Context (TLS configuration) for connection with the remote host. By default, no value is defined. To configure TLS Contexts, see Configuring TLS Certificates on page 158. **Note:** The parameter is applicable only if the connection is HTTPS.
'Verify Certificate' rest-verify-certificates [HTTPRemoteServices_ VerifyCertificate]	Enables certificate verification when connection with the host is based on HTTPS. ■ [0] **Disable** = (Default) No certificate verification is done. ■ [1] **Enable** = The device verifies the authentication of the certificate received from the HTTPS peer. The device authenticates the certificate against the trusted root certificate store associated with the associated TLS Context (see 'TLS Context' parameter above) and if ok, allows communication with the HTTPS peer. If authentication fails, the device denies communication (i.e., handshake fails). The device can also authenticate the certificate by querying with an Online Certificate Status Protocol (OCSP) server whether the certificate has been revoked. This is also configured for the associated TLS Context. **Note:** The parameter is applicable only if the connection is HTTPS.
'Verify Certificate Subject Name' verify-cert-subject-name [HTTPRemoteServices_ VerifyCertificateSubjectName]	Enables the verification of the TLS certificate subject name (Common Name / CN or Subject Alternative Name / SAN) when connection with the host is based on HTTPS. ■ [0] **Off** = (Default) No verification is done.

Parameter	Description
	■ [1] **On** = The device verifies the subject name of the certificate received from the HTTPS peer. If the server's URL contains a hostname, it verifies the certificate against the hostname; otherwise, it verifies the certificate against the server's IP address. If authentication fails, the device denies communication (i.e., handshake fails). **Note:** The parameter is applicable only if the connection is HTTPS.
Timeouts	
'Response Timeout' `rest-timeout` [HTTPRemoteServices_TimeOut]	Defines the TCP response timeout (in seconds) from the remote host. If one of the remote hosts does not respond to a request (e.g., HTTP GET method) within the specified timeout, the device closes the corresponding socket and attempts to connect to the next remote host. The valid value is 1 to 65535. The default is 5. **Note:** The global parameter for response timeout is described in Configuring a Routing Response Timeout on page 741.
'Keep-Alive Timeout' `rest-ka-timeout` [HTTPRemoteServices_ KeepAliveTimeOut]	Defines the duration/timeout (in seconds) in which HTTP-REST keep-alive messages are sent by the device if no other messages are sent. Keep-alive messages may be required for HTTP services that expire upon inactive sessions. For Remote Web Service whose 'Type' is **Routing**, **Call Status**, **Topology Status**, or **QoS**, proprietary keep-alive messages are sent. For 'Type' that is **General**, HTTP OPTIONS keep-alive messages are sent. The valid value is 0 to 65535. The default is 0 (i.e., no keep-alive messages are sent).
Status	
'Status'	(Read-only) Displays the status of the host associated with the Web service. ■ "Connected": At least one of the hosts is connected. ■ "Disconnected": All hosts are disconnected.

Parameter	Description
'Active Group'	(Read-only) Displays the currently active Group (by ID) that is associated with the Web service. This is the host group to where the device is currently sending traffic.
'Active Host'	(Read-only) Displays the currently active host (by name) that is associated with the Web service. This is the host (within the active group) to where the device is currently sending traffic. **Note:** If traffic is sent to the hosts in a round-robin fashion (i.e., 'Policy in Group' parameter is configured to **Round Robin**), then this field displays "NA".

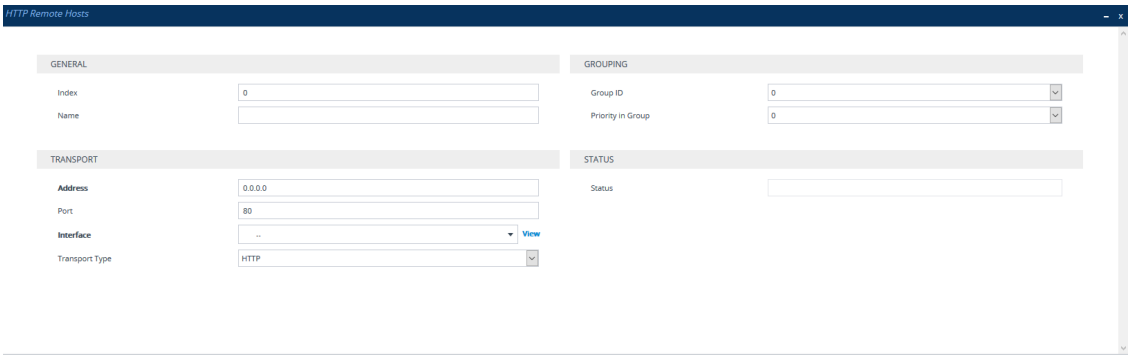
Configuring Remote HTTP Hosts

The HTTP Remote Hosts table lets you configure up to 10 remote HTTP hosts per Remote Web Service. The HTTP Remote Hosts table is a "child" of the Remote Web Services table (configured in Configuring Remote Web Services).

The following procedure describes how to configure HTTP Remote hosts through the Web interface. You can also configure it through ini file [HTTPRemoteHosts] or CLI (`configure system > http-services > http-remote-hosts`).

➢ To configure a remote HTTP host:

1. Open the Remote Web Services table (**Setup** menu > **IP Network** tab > **Web Services** folder > **Remote Web Services**).
2. In the table, select the required remote Web service index row, and then click the **HTTP Remote Hosts** link located below the table; the HTTP Remote Hosts table appears.
3. Click **New**; the following dialog box appears:



4. Configure an HTTP remote host according to the parameters described in the table below.
5. Click **Apply**, and then save your settings to flash memory.

Table 16-19:HTTP Remote Hosts Table Parameter Descriptions

Parameter	Description
General	
'Index' rest-servers [HTTPRemoteHosts_RemoteHostindex]	Defines an index number for the new table row. **Note:** ■ Each row must be configured with a unique index. ■ The parameter is mandatory.
'Name' [HTTPRemoteHosts_Name]	Defines a descriptive name, which is used when associating the row in other tables. The valid value is a string of up to 40 characters. By default, no value is defined. **Note:** ■ Each row must be configured with a unique name. ■ The parameter is mandatory.
Transport	
'Address' rest-address [HTTPRemoteHosts_Address]	Defines the address (IP address or FQDN) of the remote host. The valid value is a string of up to 80 characters. **Note:** ■ If the address is an FQDN and the DNS resolution results in multiple IP addresses, the device attempts to establish multiple connections (sessions) for each IP address. Only the first 10 resolved IP addresses are used regardless of the number of hosts. ■ FQDN resolution is also performed (immediately) when connection is subsequently "closed" (by timeout or by the remote host) and connections are updated accordingly. In addition, the device periodically (every 15 minutes) performs DNS name resolution to ensure that the list of resolved IP addresses has not changed. If a change is detected, the device updates its' list of IP addresses and re-establishes connections accordingly. ■ In addition to multiple HTTP sessions, the device establishes multiple (TCP) connections per session, thereby enhancing data exchange capabilities with the host.
'Port' rest-port	Defines the port of the host. The valid value is 0 to 65535. The default is 80.

Parameter	Description
[HTTPRemoteHosts_ Port]	
'Interface' `rest-interface` [HTTPRemoteHosts_ Interface]	Assigns one of the device's IP network interfaces through which communication with the remote host is done. By default, no value is defined and the OAMP interface is used.
'Transport Type' `rest-transport-type` [HTTPRemoteHosts_ HTTPTransportType]	Defines the protocol for communicating with the remote host: ■ [0] **HTTP** (default) ■ [1] **HTTPS**
Grouping	
'Group ID' `group-id` [HTTPRemoteHosts_ GroupID]	Defines the host's group ID. The group number (ID) reflects the priority of the group. The device sends traffic to host groups according to the configuration of the 'Policy between Groups' parameter in the Remote Web Services table. The valid value is 0 to 4, where 0 is the highest priority and 4 the lowest. The default is 0.
'Priority in Group' `host-priority-in-group` [HTTPRemoteHosts_ PriorityInGroup]	Defines the priority level of the host within the assigned group. The device sends traffic to hosts within the group according to the configuration of the 'Policy in Group' parameter in the Remote Web Services table. The valid value is 0 to 9, where 0 is the highest priority and 9 the lowest. The default is 0. **Note:** If you have configured multiple hosts in the group with the same priority, their priority is determined by their order of appearance in the table. For example, if two hosts are configured in rows Index 0 and Index 1 with priority 0, the host in Index 0 is considered higher priority.
Status	
'Status'	(Read-only) Displays the status of the connection with the remote host. ■ "Connected": The host is connected. ■ "Disconnected": The host is disconnected.

Enabling Topology Status Services

You can enable the device to send device configuration (topology) status (add, edit and delete) for Web-based services (Remote Web Services). Once enabled, you need to add a Remote Web Service with the 'Type' parameter configured to **Topology Status** (see Configuring Remote Web Services).

➢ **To enable Topology Status services:**

1. Open the Web Service Settings page (**Setup** menu > **IP Network** tab > **Web Services** folder > **Web Service Settings**).
2. From the 'Topology Status' drop-down list [RoutingServerGroupStatus], select **Enable**:

3. Click **Apply**.

Enabling Registration Status Services

You can enable the device to periodically synchronize its registration database of SIP user agents (endpoints) with a third-party Routing server (Remote Web Service). The Routing server can then use this information for routing decisions. Once enabled, you need to add a Remote Web Service with the 'Type' parameter configured to **Registration Status** (see Configuring Remote Web Services).

➢ **To enable Registration Status services:**

1. Open the Web Service Settings page (**Setup** menu > **IP Network** tab > **Web Services** folder > **Web Service Settings**).
2. From the 'Routing Server Registration Status' drop-down list [RoutingServerRegistrationStatus], select **Enable**:

3. Click **Apply**.

Third-Party Routing Server or AudioCodes Routing Manager

You can employ a remote, third-party Routing server to handle call routing decisions in deployments consisting of multiple AudioCodes devices. The Routing server can be used to handle SBC, Tel-to-IP, and IP-to-Tel calls. Employing a Routing server replaces the need for the device's routing tables--IP-to-IP Routing table for SBC calls--to determine call destination.

For more information on ARM, refer to the *ARM User's Manual* and *ARM Installation Manual*, which can be downloaded from AudioCodes website.

For SBC calls, when the device receives an incoming call (SIP INVITE, NOTIFY or MESSAGE), it searches the IP-to-IP Routing table for a matching routing rule. If the routing rule is configured to use a Routing server ('Destination Type' parameter is configured to **Routing Server**), the device sends a request to the routing server for an appropriate destination.

The request is sent to the Routing server using an HTTP Get Route message. The request contains information about the call (SIP message).

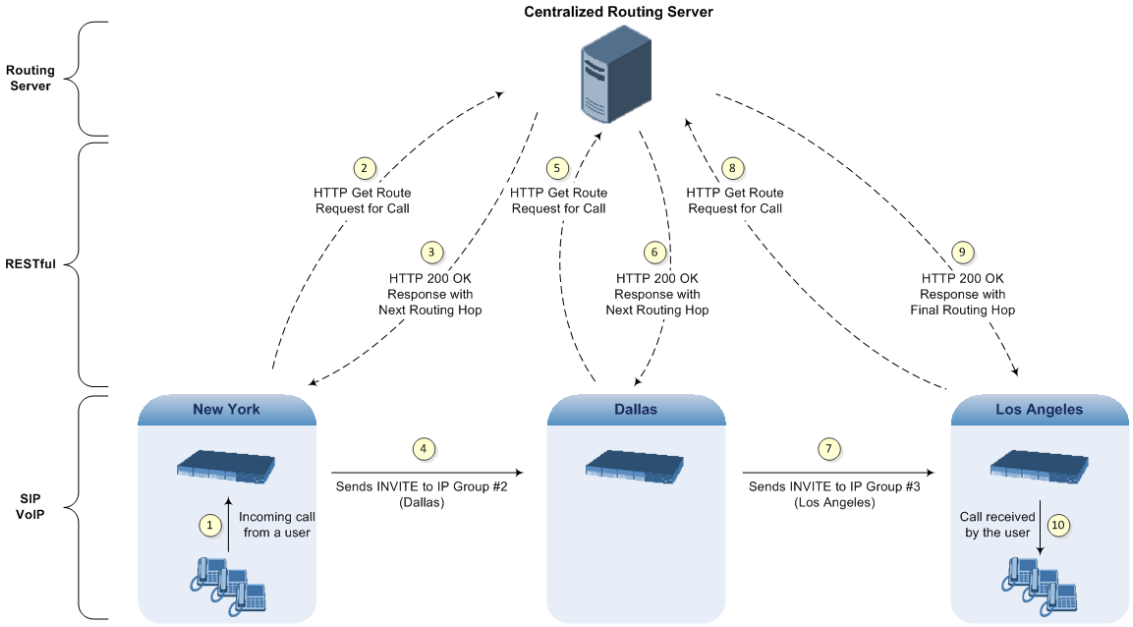
The Routing server uses its own algorithms and logic in determining the best route path. The Routing server manages the call route between devices in "hops", which may be spread over different geographical locations. The destination to each hop (device) can be by IP address (with port) or IP Group. If the destination is an IP address, even though the destination type (in the IP-to-IP Routing table) is an IP Group, the device only uses the IP Group for profiling (i.e., associated IP Profile etc.). If multiple devices exist in the call routing path, the Routing server sends the IP address only to the last device ("node") in the path.

Once the device receives the resultant destination hop from the routing server, it sends the call to that destination. The routing server can provide the device with an appropriate route or reject the call. However, if for the **initial** request (first sent Get Route request for the call), the routing server cannot find an appropriate route for the call or it does not respond, for example, due to connectivity loss (i.e., the routing server sends an HTTP 404 "Not Found" message), the device routes the call using its routing tables. If the Get Route request is not the first one sent for the call (e.g., in call forwarding or alternative routing) and the routing server responds with an HTTP 404 "Not Found" message, the device rejects the call.

This HTTP request-response transaction for the routing path occurs between Routing server and each device in the route path (hops) as the call traverses the devices to its final destination. Each device in the call path connects to the Routing server, which responds with the next hop in the route path. Each device considers the call as an incoming call from an IP Group. The session ID (SID) is generated by the first device in the path and then passed unchanged down the route path, enabling the Routing server to uniquely identify requests belonging to the same call session.

Communication between the device and routing server is through the device's embedded Representational State Transfer (RESTful) API. The RESTful API is used to manage the routing-related information exchanged between the routing server (RESTful server) and the device (RESTful client). When you have configured the device with connection settings of the routing sever and the device starts-up, it connects to the routing server and activates the RESTful API, which triggers the routing-related API commands.

The following figure provides an example of information exchange between devices and a routing server for routing calls:



The routing server can also manipulate call data such as calling name, if required. It can also create new IP Groups and associated configuration entities, if necessary for routing. Multiple routing servers can also be employed, whereby each device in the chain path can use a specific routing server. Alternatively, a single routing server can be employed and used for all devices ("stateful" routing server).

The device automatically updates (sends) the Routing server with its' configuration topology regarding SIP routing-related entities SRDs, SIP Interfaces, and IP Groups) that have been configured for use by the Routing server. For example, if you add a new IP Group and enable it for use by the Routing server, the device sends this information to the Routing server. Routing of calls associated with routing-related entities that are disabled for use by the Routing server (default) are handled only by the device (not the Routing server).

In addition to regular routing, the routing server also supports the following:

- **Alternative Routing:** If a call fails to be established, the device "closest" to the failure and configured to send "additional" routing requests (through REST API - "additionalRoute" attribute in HTTP Get Route request) to the routing server, sends a new routing request to the routing server. The routing server may respond with a new route destination, thereby implementing alternative routing. Alternatively, it may enable the device to return a failure response to the previous device in the route path chain and respond with an alternative route to this device. Therefore, alternative routing can be implemented at any point in the route path. If the routing server sends an HTTP 404 "Not Found" message for an alternative route request, the device rejects the call. If the routing server is configured to handle alternative routing, the device does not make any alternative routing decisions based on its alternative routing tables.

  If the device sends an HTTP Get Route request and the Routing server responds with a REST API attribute "action" that is set to the value 'continue', the device routes the call using its IP-to-IP Routing table. It uses the routing rule located after the original routing rule used to query the routing server ('Destination Type' set to **Routing Server**) whose 'Alternative Route Options' parameter is configured to **Route Row**. This routing can be used at any

stage of the call (e.g., after alternative routing failure by the routing server or after receiving a REFER/3xx).

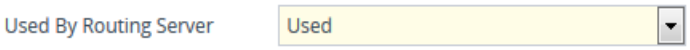
- **Call Forking:** The device can fork calls according to the routing server. When the device finds a matching routing rule in the IP-to-IP Routing table that is configured with the **Routing Server** destination, it sends an HTTP Get Route request to the routing server. When it receives a successful response from the server, the device sends an INVITE message to a destination based on the response. If the routingMethod in the response from the routing server is "fork", the device sends another HTTP Get Route request to the server and upon a successful response, sends another INVITE to another destination based on the response, and so on. This call forking process continues until no routingMethod is received from the server or it is set to “seq”, or there is a failed response from the server. If all the contacts fail (4xx), the device falls back to an alternative route, if exists, from the routing server. If 3xx is received for any of the forked destinations, the device handles it after all the forked INVITEs have been terminated.
- **Call Status:** The device can report call status to the Routing server to indicate whether a call has successfully been established and/or failed (disconnected). The device can also report when an IP Group (Proxy Set) is unavailable, detected by the keep-alive mechanism, or when the CAC thresholds permitted per IP Group have been crossed.
- **Credentials for Authentication:** The routing server can provide user (e.g., IP Phone caller) credentials (username-password) in the Get Route response, which can be used by the device to authenticate outbound SIP requests if challenged by the outbound peer, for example, Microsoft Skype for Business (per RFC 2617 and RFC 3261). If multiple devices exist in the call routing path, the routing server sends the credentials only to the last device ("node") in the path.

  Alternatively, the device can authenticate incoming SIP requests (INVITE or REGISTER) from User-type IP Groups, by first obtaining (REST-based API query) the user's password from the routing server where it is stored. When this feature is enabled and the device receives an incoming SIP dialog-initiating request, it sends the REST API command getCredentials in the Get request to the routing server. The name of the user whose credentials are requested is obtained from the SIP From header when authenticating an INVITE message, and from the To header when authenticating a REGISTER message. The routing server sends a 200 response to the device containing the password (if the requested user exists). The device then sends the challenge back to the user. The user resends the request with a SIP Authorization header (containing a response to the challenge), and the authentication process continues in the usual manner. If the device doesn’t receive a password, it rejects the incoming dialog (SIP 404). To enable this authentication type, you need to configure the IP Group's 'SBC Server Authentication Type' parameter to **ARM Authentication** (see Configuring IP Groups on page 451). Note that the routing server does not authenticate users, but only helps the device to process the SIP Digest authentication by providing the user credentials.
- **QoS:** The device can report QoS metrics per IP Group to the routing server which it can use to determine the best route (i.e., QoS-based routing). For more information, see Configuring QoS-Based Routing by Routing Server.

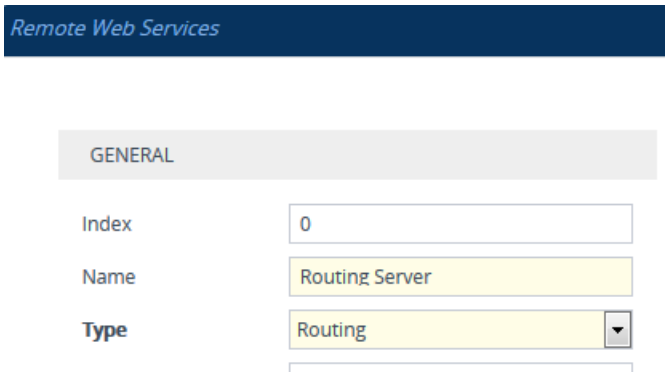
- **Call Preemption for Emergency Calls:** If you enable call preemption for emergency calls (e.g., 911) on the device, the routing server determines whether or not the incoming call is an emergency call and if so, handles the routing decision accordingly (i.e., preempts a non-emergency call if the maximum call capacity of the device is reached in order to allow the emergency call to be routed). To enable call preemption for emergency calls, use the [SBCPreemptionMode] parameter .
- **Registration status:** The device can periodically synchronize its registration database of SIP user agents (endpoints) with the routing server to keep it up to date, enabling the routing server to use this information to perform correct and optimal routing decisions. To enable this functionality, see Enabling Registration Status Services on page 328.

➢ **To configure routing based on Routing server:**

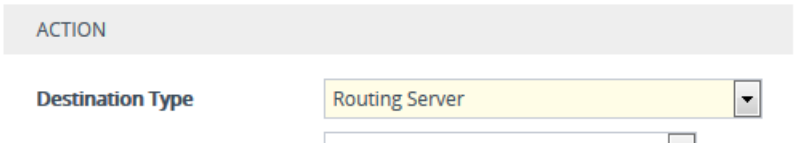
1. For each configuration entity (e.g., IP Group) that you want routing done by the routing server, configure the entity's 'Used By Routing Server' parameter to **Used**:



2. Configure an additional Security Administrator user account in the Local Users table (see Configuring Management User Accounts) that is used by the routing server (REST client) to log in to the device's management interface.
3. Configure the address and connection settings of the routing server, referred to as a *Remote Web Service* and an HTTP remote host (see Configuring Remote Web Services). You must configure the 'Type' parameter of the Remote Web Service to **Routing**, as shown in the example:



4. In the IP-to-IP Routing table, configure the 'Destination Type' parameter of the routing rule to **Routing Server** (see Configuring SBC IP-to-IP Routing Rules):



Configuring QoS-Based Routing by Routing Server

You can configure the device to allow the routing server to route calls based on QoS metrics (media and signaling). The device collects QoS metrics per IP Group that you have configured to operate with the routing server ('Used by Routing Server' parameter configured to **Used** in the IP Groups table). The metrics include the following:

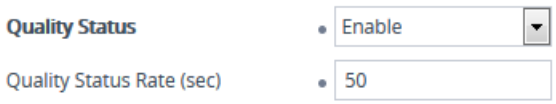
- **Signaling:** ASR, NER, and ACD
- **Media:** Packet loss (Rx/Tx), packet delay (local/remote), jitter (local/remote), MOS (local/remote), audio bandwidth (Rx/Tx), video bandwidth (Rx/Tx), and total bandwidth (Rx/Tx)

The device collects QoS metrics for both incoming call traffic and outgoing traffic from the remote endpoint. It sends the QoS reports to the routing server, where each report can contain the status of up to 100 IP Groups. If more than 100 IP Groups exist, the device sends multiple QoS reports (sequentially) to the routing server. The device sends the reports every user-defined period. The routing logic of where to route calls based on QoS ("good", "fair", and "bad") is configured on the routing server.

> For media metrics calculations, the device's License Key must include voice quality monitoring and RTCP XR.

➢ **To configure the device for QoS-based routing by routing server:**

1. Open the Web Service Settings page (**Setup** menu > **IP Network** tab > **Web Services** folder > **Web Service Settings**), and then do the following:
   a. From the 'Quality Status' [RoutingServerQualityStatus] drop-down list, select **Enable** to enable QoS-based routing.
   b. In the 'Quality Status Rate' field (RoutingServerQualityStatusRate), enter the rate (in sec) at which the device sends QoS reports.



   c. Click **Apply**.
2. Open the Remote Web Services table (see Configuring Remote Web Services), and then for the Remote Web Service entry that you configured for the routing server, do the following:
   a. From the 'Type' [HTTPRemoteServices_HTTPType] drop-down list, select **QoS**.
   b. Click **Apply**.
3. Enable voice quality monitoring and RTCP XR, using the 'Enable RTCP XR' [VQMonEnable] parameter (see Configuring RTCP XR).

Configuring an HTTP GET Web Service

You can query (HTTP GET) an HTTP server and use the response for various functionality such as routing or saving it, for example, as a session variable in order to use it in SIP message manipulations.

You need to configure a Remote Web Service to represent the HTTP server and a Call Setup Rule to define the search query and the action you want done based on the HTTP response. The following example queries an HTTP server (at IP address 52.7.189.10) using the caller's (source) user name in the server's path */v3/phone*. When a response is received from the HTTP server, the device adds the value of the HTTP response body ("Alice") to the From header in the outgoing SIP message.

➢ **To configure an HTTP GET operation:**

1. Open the Remote Web Services table, and then configure a Remote Web Service for the HTTP server:
   - 'Name': **MyHTTP**
   - 'Type': **General**
   - 'Path': **v3/phone**
   - 'Username': **adminuser1**
   - 'Password': **1234**
2. Open the HTTP Remote Hosts table of the Remote Web Service that you configured in Step 1, and then configure the following:
   - 'Name': **MyHTTPHost**
   - 'Address': **52.7.189.10**
3. Open the Call Setup Rules table, and then configure the following rule:
   - 'Rule Set ID': **1**
   - 'Request Type': **HTTP GET**
   - 'Request Target': **MyHTTP**
   - 'Request Key': **Param.Call.Src.User+'?account_sid=SID&auth_token=TOKEN'**
   - 'Action Subject': **Param.Call.Src.Name**
   - 'Action Type': **Modify**
   - 'Action Value': **HTTP.Response.Body**
4. Assign your Call Setup Rule to the relevant SIP Interface, for example.

An example of the HTTP and SIP messages of the above configuration is shown below:

1. Incoming SIP message:

```
INVITE sip:2000@10.7.7.246;user=phone SIP/2.0
Via: SIP/2.0/UDP 10.7.2.15;branch=z9hG4bKLRGQTOQHILSSMGAQJQSU
From: <sip:
15551234567
@10.7.2.15;user=phone>;tag=RJFNXMKDOHELDUMEWWGH
To: <sip:2000@10.7.7.246;user=phone>
Call-ID: UBBKFKBCXFPESMYOPDTB@10.7.2.15
CSeq: 1 INVITE
Contact: <sip:1000@10.7.2.15>
Supported: em,100rel,timer,replaces
Allow:
REGISTER,OPTIONS,INVITE,ACK,CANCEL,BYE,NOTIFY,PRACK,REFER,I
NFO,SUBSCRIBE
User-Agent: Sip Message Generator V1.0.0.5
```

2. Outgoing HTTP GET:

```
Header=GET /v3/phone/15551234567?account_sid=SID&auth_
token=TOKEN HTTP/1.1
Content-Type: html/text
Host: 52.7.189.114
Connection: keep-alive
Content-Length: 0
Cache-Control: no-cache
User-Agent: 1
```

3. Incoming HTTP response:

```
HTTP/1.1 200 OK
Access-Control-Allow-Origin: *
Cache-Control: max-age=0
Content-Type: text/html
Date: Thu, 07 Dec 2017 14:35:21 GMT
Server: nginx/1.8.1
Content-Length: 6
Connection: keep-alive

Alice
```

4. Outgoing SIP message:

```
INVITE sip:2000@10.7.7.246;user=phone SIP/2.0
Via: SIP/2.0/UDP 10.7.7.246:5060;branch=z9hG4bKac1693897511
```

```
Max-Forwards: 70
From: Alice
<sip:+15551234567@10.7.2.15;user=phone>;tag=1c1900944531
To: <sip:2000@10.7.7.246;user=phone>
Call-ID: 176518124411201016 54@10.7.7.246
CSeq: 1 INVITE
Contact: <sip:1000@10.7.7.246:5060>
Supported: em,100rel,timer,replaces,sdp-anat
```

Configuring HTTP POST Web Service

You can use HTTP POST messages to simply notify an HTTP server about a call, and use HTTP POST messages for querying information like HTTP GET messages. The example provided in this section describes how to configure the device to send HTTP POSTs to notify an HTTP server of incoming 911 calls. You need to configure a Remote Web Service to represent the HTTP server (IP address 52.7.189.10). You also need to configure Call Setup Rules that instruct the device to send an HTTP POST message, containing the 911 caller's user name and host name, to the server (at path */path/query/notify-emergency-call*) if a 911 call is received.

➢ **To configure an HTTP POST notification operation:**

1. Open the Remote Web Services table, and then configure a Remote Web Service for the HTTP server:
   - 'Name': **MyHTTP**
   - 'Type': **General**
   - 'Username': **adminuser1**
   - 'Password': **1234**
2. Open the HTTP Remote Hosts table of the Remote Web Service that you configured in Step 1, and then configure the following:
   - 'Name': **MyHTTPHost**
   - 'Address': **52.7.189.10**
3. Open the Call Setup Rules table, and then configure the following rules (Rule Set ID 1):
   - If the destination number of the incoming call is not 911, then don't process these Call Setup Rules:
     - 'Index': **1**
     - 'Rule Set ID': **1**
     - 'Condition': **Param.Call.Dst.User != '911'**
     - 'Action Type': **Exit**
     - 'Action Value': **True**

- Set the Content-Type header in the HTTP POST message to the value "application/json":
  - 'Index': **2**
  - 'Rule Set ID': **1**
  - 'Action Subject': **HTTP.Request.Content-Type**
  - 'Action Type': **Modify**
  - 'Action Value': **'application/json'**
- Add JSON parameters to the body of the HTTP POST message so that it includes the 911 caller's (source) number and host name:
  - 'Index': **3**
  - 'Rule Set ID': **1**
  - 'Action Subject': **HTTP.Request.Body**
  - 'Action Type': **Add**
  - 'Action Value': **'{ "user": "'+Param.Call.Src.User+'", "host": "'+Param.Call.Src.Host+'" }'**
- Send the HTTP POST message to the specified server and folder path:
  - 'Index': **4**
  - 'Rule Set ID': **1**
  - 'Request Type': **HTTP POST Notification**
  - 'Request Target': **MyHTTP**
  - 'Request Key': **'/path/query/notify-emergency-call'**

4. Assign your Call Setup Rules (i.e., Rule Set ID 1) to the relevant SIP Interface (for example).

An example of the HTTP and SIP messages of the above configuration is shown below:

1. Incoming SIP message from 911 caller:

```
INVITE sip:911@10.7.7.246;user=phone SIP/2.0
Via: SIP/2.0/UDP 10.7.2.15;branch=z9hG4bKLRGQTOQHILSSMGAQJQSU
From: <sip:
15551234567@10.7.2.15;user=phone>;tag=RJFNXMKDOHELDUMEWWGH
To: <sip:911@10.7.7.246;user=phone>
Call-ID: UBBKFKBCXFPESMYOPDTB@10.7.2.15
CSeq: 1 INVITE
Contact: <sip:1000@10.7.2.15>
Supported: em,100rel,timer,replaces
Allow:
```

```
REGISTER,OPTIONS,INVITE,ACK,CANCEL,BYE,NOTIFY,PRACK,REFER,I
NFO,SUBSCRIBE
User-Agent: Sip Message Generator V1.0.0.5
```

2. Outgoing HTTP POST message notifying server of 911 call:

```
Header=POST /path/query/notify-emergency-call HTTP/1.1
Content-Type: application/json
Host: 52.7.189.114
Connection: keep-alive
Content-Length: 47
Cache-Control: no-cache
User-Agent: 1

{ "user": "15551234567", "host": "10.7.2.15" }
```

Configuring Web Service for Automatic Provisioning

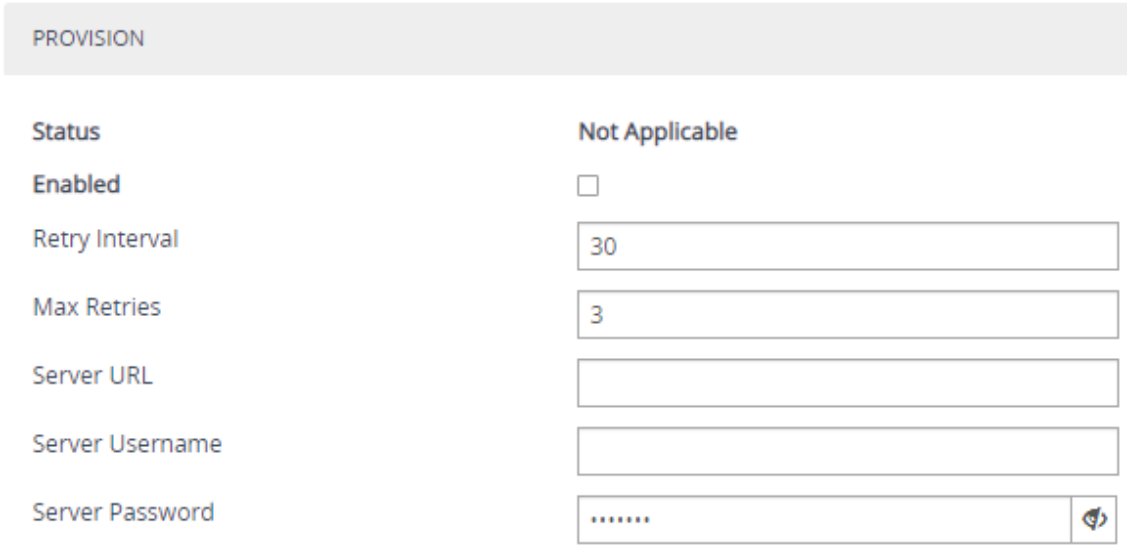
You can configure a Remote Web Service for automatic provisioning of the device by a remote HTTP server. For this feature, whenever the device resets or powers up, the device uses REST API to send an HTTP/S POST request with JSON content to the server with identification information (e.g., serial number). If the server identifies the device and has an updated configuration file, it transfers the file to the device using Secure Copy Protocol (SCP).

If the request fails (any HTTP response other than 200 OK), the device sends another request after a user-defined time. The maximum number of retries is three. If it still fails, the device's **Status** LED located on the chassis blinks green and the 'Status' field (see below) displays "Operation Failed".

- The downloaded configuration file (CLI Script file) overwrites the existing configuration.

➢ **To configure automatic provisioning as a Remote Web Service:**

1. Open the Web Service Settings page (**Setup** menu > **IP Network** tab > **Web Services** folder > **Web Service Settings**).
2. Scroll down to the **Provision** group:



3. Select the 'Enabled' check box to enable the provisioning feature.
4. In the 'Retry Interval' field, configure the time (in seconds) between each sent HTTP request that failed.
5. In the 'Max Retries' field, configure the maximum number of attempts to send the request before provisioning is considered a failure.
6. In the 'Server URL' field, configure the provisioning server's path where the requests must be sent.
7. In the 'Server Username' and 'Server Password' fields, configure the username and password respectively for authentication with the server.
8. Click **Apply**, and then reset the device with a burn-to-flash for your settings to take effect.

The 'Status' field displays the status of the automatic provisioning:

- "Not Applicable": Automatic provisioning is disabled or the device doesn't support the provisioning process.
- "Initialization Failed": Typically, some preliminary initialization failed (e.g., couldn't create HTTP service).
- "In Progress": The device is currently sending the HTTP request.
- "Resolving DNS": If the URL contains an FQDN, it is currently being resolved into an IP address by a DNS server.
- "Wrong Server URL": The configured URL is incorrect.
- "Bad Response": The sent HTTP request failed and the device is making another attempt.
- "Operation Failed": The number of request attempts have exceeded the number of configured retries.
- "Operation Succeeded": A 200 OK HTTP response has been received from the provisioning server.

HTTP-based Proxy Services

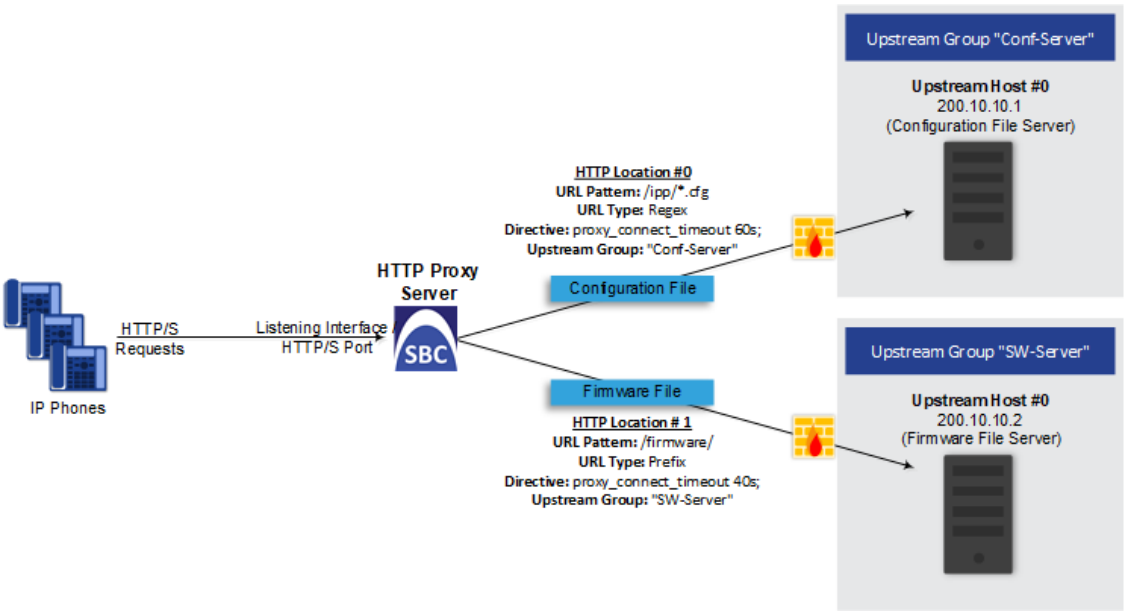
The device can be configured as an HTTP Proxy (or a non-HTTP Proxy) to intermediate between clients (e.g., HTTP requests) and servers (hosts). The client sends an HTTP GET request specifying the destination (URL), and the device (acting as an HTTP Proxy), forwards it to the host, and vice versa.

This feature integrates the NGINX platform, which is an open source proxy server. When you enable the HTTP Proxy application, the NGINX daemon is launched. When you then configure the various HTTP Proxy tables, the configuration is reflected in the NGINX configuration file (nginx.conf). Each parameter has its own NGINX Directive notation syntax, which is shown in the nginx.conf file. For example, if you have configured an Upstream Host (discussed later in this section), it is displayed in the NGINX file as "server host:port". You can also configure and apply additional NGINX Directives that do not have any corresponding parameters in the device's Web interface. For more information, see Configuring HTTP Directive Sets on page 360.

The device supports the following proxy services:

- **HTTP Reverse Proxy for managing equipment behind NAT:** You can configure the device to function as a Reverse HTTP Proxy server. You can use this for many HTTP-based applications. For example, you can use it to intermediate between REST API clients and a REST server.

  Another example is to use the HTTP Proxy to intermediate between IP Phones and remote servers for file download. The figure below illustrates this example where IP Phones (clients) retrieve their configuration and firmware files from remote file servers (Upstream Hosts) and where the device (HTTP Proxy) intermediates between the two. The HTTP hosts are located in the cloud and the clients are located behind NAT. The HTTP Proxy listens for incoming HTTP requests (Listening Interface and HTTP/S Listening Port) from the clients and then forwards the requests to the relevant HTTP host, based on the URL (HTTP Location) in the incoming HTTP GET request. If the URL matches the pattern "/firmware/", the HTTP Proxy sends the request to the firmware file server; if the URL matches the pattern "/ipp/*.cfg", the requests are sent to the configuration file server. In addition, customized NGINX directives have been configured for each HTTP Location to define the maximum time to wait for an HTTP connection.



A summary of the required configuration for this example is listed below:

a. Enable the HTTP Proxy application (see Enabling the HTTP Proxy Application on the next page).

b. Configure two Upstream Groups, where each is configured with an Upstream Host that defines the IP address of the HTTP host (i.e., firmware and configuration file servers). See Configuring Upstream Groups on page 356.

c. Configure two NGINX directives for proxy timeout connection (see Configuring HTTP Directive Sets on page 360).

d. Configure a local, listening IP network interface for the leg interfacing with the HTTP clients (see Configuring IP Network Interfaces on page 124) or use the default.

e. Configure a local, IP network interface for the outbound leg interfacing with the HTTP hosts (or use the default).

f. Configure the HTTP Proxy server, by assigning it the listening IP network interface and configuring a listening HTTP/S port (see Configuring HTTP Proxy Servers on page 343).

g. Configure two HTTP Locations for the HTTP Proxy server, where each is configured with a URL pattern to match the incoming HTTP requests for determining the destination host (Upstream Group-Upstream Host). In addition, assign it the relevant HTTP Directive Set. See Configuring HTTP Locations on page 348.

- **Non-HTTP Proxy (referred to as TCP/UDP Proxy Server):** The device can serve as a proxy for other applications that are not based on HTTP. For example, it can be used to intermediate between clients and a DNS server for DNS lookup, or between clients and an NTP server for clock synchronization. For more information, see Configuring TCP-UDP Proxy Servers on page 352.

- **HTTP-based OVOC service for AudioCodes equipment located behind NAT that are managed by the AudioCodes OVOC server:** For more information, see Configuring an HTTP-based OVOC Service on page 363

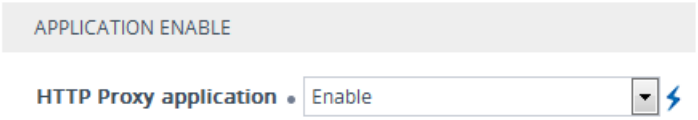
Enabling the HTTP Proxy Application

Before you can configure HTTP-based proxy services, you must enable the HTTP Proxy application, as described in the following procedure. Once enabled, the Web interface displays menus in the Navigation pane that are relevant to the HTTP Proxy application.

> The HTTP Proxy application is a license-based feature and is available only if it is included in the License Key installed on the device. For ordering the feature, please contact the sales representative of your purchased device. For installing a new License Key, see License Key.

➢ **To enable the HTTP Proxy application:**

1. Open the General Settings page (**Setup** menu > **IP Network** tab > **HTTP Proxy** folder > **General Settings**).



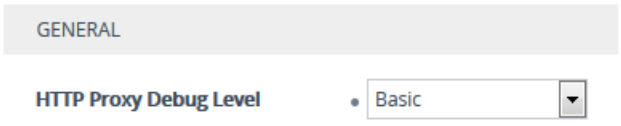
2. From the 'HTTP Proxy Application' drop-down list, select **Enable**.
3. Click **Apply**, and then reset the device with a save-to-flash for your settings to take effect.

Debugging Remote HTTP Services

You can enable the device to generate debug messages for remote Web (HTTP) services and have them sent to a Syslog server.

➢ **To enable debugging of HTTP services:**

1. Open the General Settings page (**Setup** menu > **IP Network** tab > **HTTP Proxy** folder > **General Settings**):



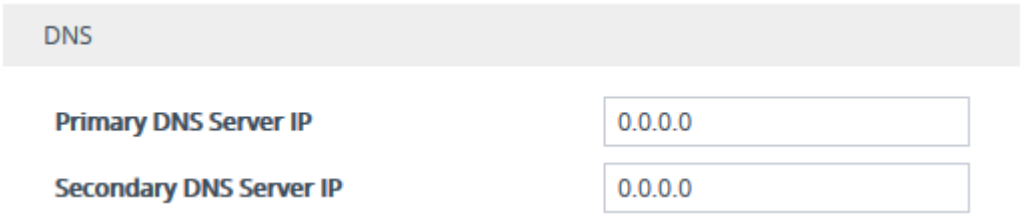
2. From the 'HTTP Proxy Debug Level' drop-down list, select a begug level.
3. Click **Apply**.

Configuring a DNS Server for HTTP Services

You can configure the DNS server (primary and secondary for redundancy) that you want to use for your HTTP services. If you configure a proxy server with an FQDN, this is the DNS server that the device uses for resolving the domain name into an IP address.

➢ **To configure DNS servers for HTTP services:**

1. Open the General Settings page (**Setup** menu > **IP Network** tab > **HTTP Proxy** folder > **General Settings**):



2. In the 'Primary DNS Server IP' field, enter the IP address of your main DNS server.
3. (Optional) In the 'Secondary DNS Server IP' field, enter the IP address of the secondary DNS server.
4. Click **Apply**.

When generating the NGINX configuration file, the device includes the resolver directive, specifying the primary and secondary DNS servers, as configured above. However, NGINX supports optional parameters that allow you to fine-tune the behavior of the DNS resolution. You can include these additional parameters using the ini file parameter [NginxResolverParams), which is added to the resolver directive when the device generates the NGINX configuration file. For more information on these optional parameters, go to http://nginx.org/en/docs/http/ngx_ http_ core_ module.html#resolver.

Configuring HTTP Proxy Servers

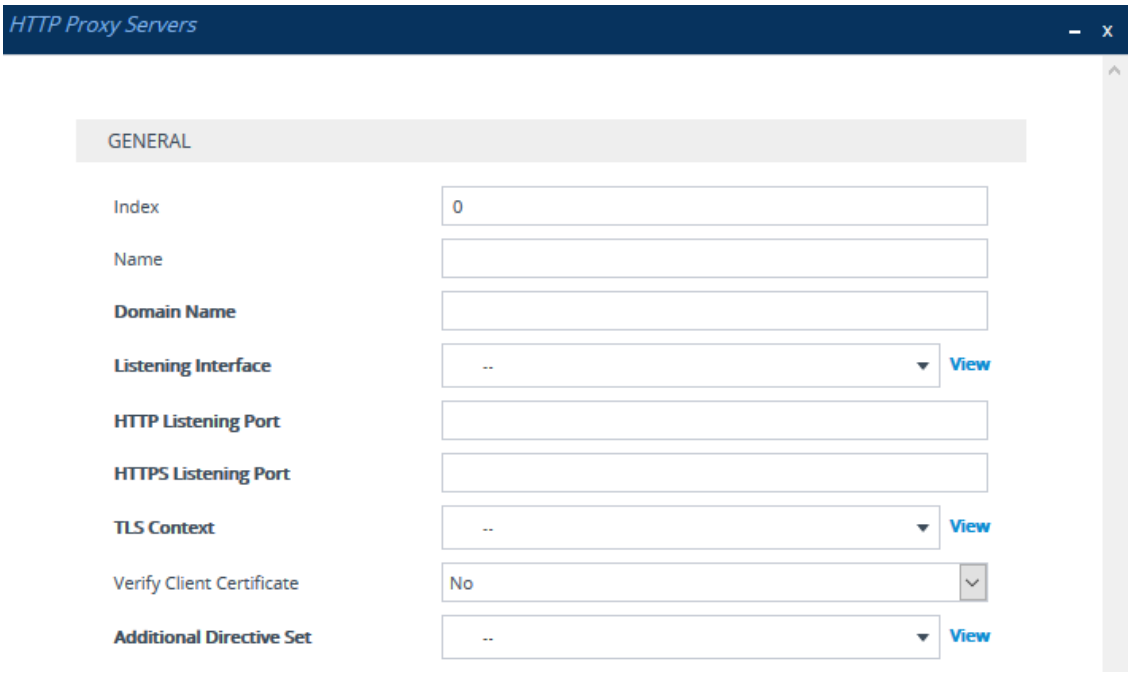
The HTTP Proxy Servers table lets you configure up to 10 HTTP Proxy servers. Once configured, you can configure HTTP Locations for the HTTP Proxy Server (see Configuring HTTP Locations on page 348).

The following procedure describes how to configure HTTP Proxy Servers through the Web interface. You can also configure it through ini file [HTTPServer] or CLI (`configure network > http-proxy > http-server`).

➢ **To configure an HTTP Proxy Server:**

1. Enable the HTTP Proxy application, as described in Enabling the HTTP Proxy Application on the previous page.

2. Open the HTTP Proxy Server table (**Setup** menu > **IP Network** tab > **HTTP Proxy** folder > **HTTP Proxy Servers**).
3. Click **New**; the following dialog box appears:



4. Configure an HTTP Proxy server according to the parameters described in the table below.
5. Click **Apply**, and then save your settings to flash memory.

Table 16-20:HTTP Proxy Servers Table Parameter Descriptions

Parameter	Description
'Index' [HTTPServer_Index]	Defines an index number for the new table row. **Note:** ■ Each row must be configured with a unique index. ■ The parameter is mandatory.
'Name' name [HTTPServer_ ServiceName]	Defines a descriptive name, which is used when associating the row in other tables. The valid value is a string of up to 40 characters. By default, no value is defined. **Note:** ■ Each row must be configured with a unique name.

Parameter	Description
	■ The parameter is mandatory. ■ The parameter value cannot contain a forward slash (/).
'Domain Name' `domain-name` [HTTPServer_ DomainName]	Defines a domain name (FQDN). This is configured only when multiple DNS domains share the same IP address. **Note:** The NGINX directive for this parameter is "server_name".
'Listening Interface' `listening-int` [HTTPServer_ ListeningInterface]	Assigns an IP Interface to the HTTP Proxy service. To configure IP Interfaces, see Configuring IP Network Interfaces on page 124. **Note:** ■ The parameter is mandatory. ■ The NGINX directive for this parameter is " listen ip".
'HTTP Listening Port' `http-port` [HTTPServer_ HTTPListeningPort]	Defines the HTTP listening port, which is the local port for incoming packets for the HTTP service. **Note:** ■ The port number must not conflict with the ports used for the Web interface, which is usually 80 for HTTP and 443 for HTTPS. ■ You must configure at least one port (HTTP or HTTPS port). ■ The NGINX directive for this parameter is "listen ip:port".
'HTTPS Listening Port' `https-port` [HTTPServer_ HTTPSListeningPort]	Defines the HTTPS listening port, which is the local port for incoming packets for the HTTP service. **Note:** ■ The port number must not conflict with the ports used for the Web interface, which is usually 80 for HTTP and 443 for HTTPS.

Parameter	Description
	■ You must configure at least one port (HTTP or HTTPS port). ■ The NGINX directive for this parameter is "listen ip:port ssl".
'TLS Context' `tls-context` [HTTPServer_TLSContext]	Assigns a TLS Context (TLS configuration). This is required if you have specified an HTTPS listening port (see the 'HTTPS Listening Port' parameter above). To configure TLS Contexts, see Configuring TLS Certificate Contexts on page 158. **Note:** The NGINX directives for this parameter is "tls-context", "ssl_certificate", "ssl_certificate_key", "ssl_ciphers", "ssl_protocols", and "ssl_password_file".
Verify Client Certificate `verify-client-cert` [HTTPServer_VerifyCertificate]	Enables the verification of the client TLS certificate, where the client is the device or user that issues the HTTPS request. ■ [0] **No** = (Default) No certificate verification is done. ■ [1] **Yes** = The device verifies the authentication of the certificate received from the HTTPS client. The device authenticates the certificate against the trusted root certificate store associated with the assigned TLS Context (see 'TLS Context' parameter above) and if ok, allows communication with the HTTPS client. If authentication fails, the device denies communication (i.e., handshake fails). The device can also authenticate the certificate by querying with an Online Certificate Status Protocol (OCSP) server whether the certificate has been revoked. This is also configured for the associated TLS Context.

Parameter	Description
'Additional Directive Set' `directive-set` [HTTPServer_ AdditionalDirectiveSet]	Assigns an NGINX Directive Set for the HTTP service. To configure HTTP Directive Sets, see Configuring HTTP Directive Sets on page 360.
'URL Prefix' `url-prefix` [HTTPProxyService_ URLPrefix]	Defines the URL prefix that is used to access the managed equipment's embedded Web server. The URL prefix is matched against the target of the HTTP requests sent by the client (such as GET and POST). If a match is located in the table, the device removes the prefix from the request and then forwards the HTTP request to the managed equipment without the prefix. For example, for the URL of GET /home/index.html HTTP/1.1 (which is part of the URL http://10.20.30.40/home/index.html), a URL prefix of "/home" can be configured. To match all URLs, configure the parameter to "/" (default).
'Keep-Alive Mode' `keep-alive-mode` [HTTPProxyService_ KeepAliveMode]	Enables a keep-alive mechanism with the managed equipment: ■ [0] **Disable** ■ [1] **Options** = (Default) Enables keep-alive by sending HTTP OPTIONS messages. If no response is received from the HTTP host, the device stops forwarding HTTP requests to the host and raises an SNMP alarm (acHTTPProxyServiceAlarm). If you configured the address of the host as an FQDN (see Configuring HTTP Proxy Hosts) and the DNS resolution results in multiple IP addresses, when no response is received from the keep-alive, the device checks connectivity with the next resolved IP address and so on, until a response is received.

Configuring HTTP Locations

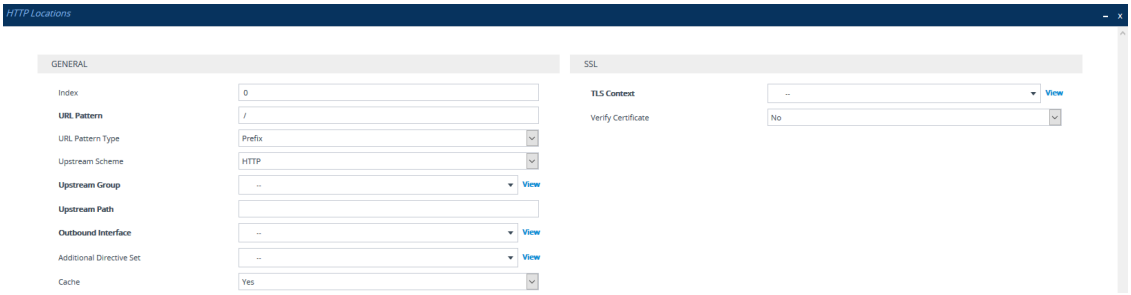
The HTTP Locations table lets you configure up to 40 HTTP Locations. Locations are specified for each HTTP proxy server to map the incoming requests received by that server. Mapping is based on matching the URL in the request. Each location specifies the URL prefix or pattern to match and the target Upstream Group to which the request is to be forwarded to.

The HTTP Locations table is a "child" of the HTTP Proxy Servers table (see Configuring HTTP Proxy Servers on page 343), reflecting the nesting of Location contexts within Server contexts in the NGINX configuration file. This may be used to specify unique handling of URLs by file type (using a regex pattern) or by pathname (using a Prefix or Exact Match pattern).

The following procedure describes how to configure HTTP Locations through the Web interface. You can also configure it through ini file [HTTPLocation] or CLI (`configure network > http-proxy > location`).

➢ To configure an HTTP Location:

1. Enable the HTTP Proxy application, as described in Enabling the HTTP Proxy Application on page 342.
2. Open the HTTP Proxy Servers table (**Setup** menu > **IP Network** tab > **HTTP Proxy** folder > **HTTP Proxy Servers**).
3. In the table, select the required HTTP Proxy Server, and then click the **HTTP Locations** link located below the table; the HTTP Locations table appears.
4. Click **New**; the following dialog box appears:



5. Configure an HTTP Location according to the parameters described in the table below.
6. Click **Apply**, and then save your settings to flash memory.

Table 16-21:HTTP Locations Table Parameter Descriptions

Parameter	Description
General	
'Index' [HTTPLocation_Index]	Defines an index number for the new table row. **Note:** ■ Each row must be configured with a unique index.

Parameter	Description
	■ The parameter is mandatory.
'URL Pattern' `url-pattern` [HTTPLocation_ URLPattern]	Defines the URL pattern. Received GET or POST requests are matched against the locations in the HTTP Locations table by matching the URL in the received request to the URL configured by this parameter. If there is a match, the prefix is stripped from the request and then forwarded in the outgoing HTTP request. **Note:** ■ The pattern must be based on the pattern type configured in the 'URL Pattern Type' parameter (see below). ■ The NGINX directive for this parameter is "location modifier pattern".
'URL Pattern Type' `url-pattern-type` [HTTPLocation_ URLPatternType]	Defines the type of URL pattern used for configuring the 'URL Pattern' parameter (see above). ■ [0] **Prefix** = For Example, "/" matches any URL beginning with a forward slash "/". For NGINX, this option has no modifier. ■ [1] **Exact** = Defines an exact pattern to match, for example, "/abc/def" matches only the file "/abc/def". For NGINX, this option is specified using the "=" modifier. ■ [2] **Regex** = Regex-based pattern (case sensitive), for example, "/files/*.img" matches all files ending in .img in the directory /files. For NGINX, this option is specified using the "~" modifier. ■ [3] **Case-Insensitive Regex** = Regex-based pattern that is case-insensitive, for example, "*.img" matches abc.IMG as well as xyz.img. For NGINX, this option is specified using the "~*" modifier. ■ [4] **Prefix Ignore Regex** = For NGINX, this option is specified using the "^~" modifier. For example, assume that you have configured the following URL patterns for four HTTP Locations: ■ 1) /files – **Prefix** pattern type ■ 2) /files/phone – **Prefix** pattern type ■ 3) /files/firmware -- **Prefix-Ignore-Regex** pattern type ■ 4) *.jpg – **Regex** pattern type

Parameter	Description
	Therefore, the request URL "/files/phone/aaa" matches Location 2 and the request URL "/files/phone/logo.jpg" matches Location 4. The request URL "/files/firmware/logo.jpg" matches Location 3 (and not Location 4). **Note:** The NGINX directive for this parameter is "location modifier pattern". For more information on NGINX modifiers, see ngx_http_core_module.html.
'Upstream Scheme' `upstream-scheme` [HTTPLocation_ UpstreamScheme]	Defines the protocol for sending requests to the Upstream Group. ■ [0] **HTTP** (default) ■ [1] **HTTPS** **Note:** The NGINX directive for this parameter is "proxy_pass scheme://upstream".
'Upstream Group' `upstream-group` [HTTPLocation_ UpstreamGroup]	Assigns a group of servers (Upstream Group) to handle the HTTP requests. To configure Upstream Groups, see Configuring Upstream Groups on page 356. **Note:** The NGINX directive for this parameter is "proxy_pass scheme://upstream".
'Upstream Path' `upstream-path` [HTTPLocation_ UpstreamPath]	Defines a path to prepend to the URL before sending the request to the Upstream Group. **Note:** The NGINX directive for this parameter is "proxy_pass scheme://upstream/path".
'Outbound Interface' `outbound-intfc` [HTTPLocation_ OutboundInterface]	Assigns a local, IP network interface for sending requests to the Upstream Group. To configure IP network interfaces, see Configuring IP Network Interfaces on page 124. By default, no value is defined. **Note:** ■ The parameter is mandatory. ■ The NGINX directive for this parameter is "proxy_bind".
'Additional Directive Set' `directive-set` [HTTPLocation_ AdditionalDirectiveSet]	Assigns an NGINX directive set for the HTTP location. To configure NGINX directives, see Configuring HTTP Directive Sets on page 360.
'Cache'	Enables the caching of files in this location.

Parameter	Description
`cache` [HTTPLocation_Cache]	■ [0] **No** ■ [1] **Yes** (default) **Note:** ■ Currently, this feature is not supported. ■ The NGINX directive for this parameter is "proxy_cache zone|off".
SSL	
'TLS Context' `tls-context` [HTTPLocation_ TLSContext]	Assigns a TLS Context for the TLS connection with the HTTP location. To configure TLS Contexts, see Configuring TLS Certificates on page 158. **Note:** ■ The parameter is applicable only if the connection protocol is HTTPS (configured in the 'Upstream Scheme' parameter, above). ■ The NGINX directives for this parameter are "proxy_ssl_ certificate", "proxy_ssl_certificate_key", "proxy_ssl_ ciphers", "proxy_ssl_protocols", and "proxy_ssl_ password_file".
'Verify Certificate' `verify-cert` [HTTPLocation_ VerifyCertificate]	Enables TLS certificate verification when the connection with the location is based on HTTPS. It verifies the certificate of the incoming connection request from the Upstream Group. ■ [0] **No** = (Default) No certificate verification is done. ■ [1] **Yes** = The device verifies the authentication of the certificate received from the HTTPS location. The device authenticates the certificate against the trusted root certificate store associated with the assigned TLS Context (see 'TLS Context' parameter above) and if ok, allows communication with the HTTPS location. If authentication fails, the device denies communication (i.e., handshake fails). The device can also authenticate the certificate by querying with an Online Certificate Status Protocol (OCSP) server whether the certificate has been revoked. This is also configured for the associated TLS Context. **Note:** ■ The parameter is applicable only if the connection

Parameter	Description
	protocol is HTTPS (configured in the 'Upstream Scheme' parameter, above). ■ The NGINX directive for this parameter is "proxy_ssl_ verify".

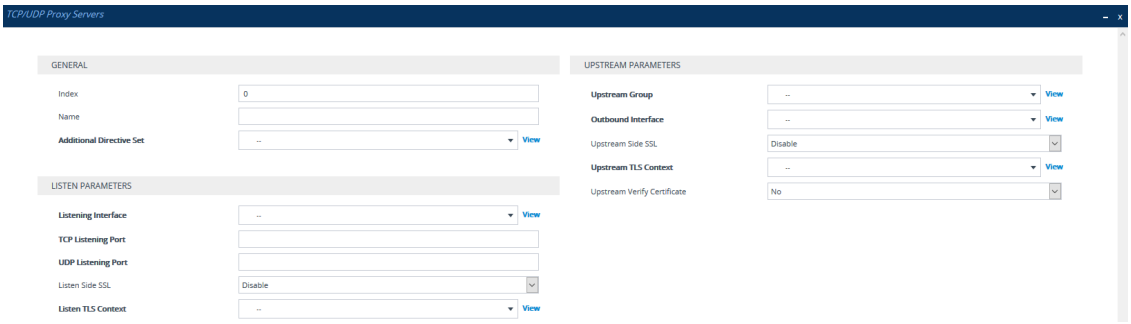
Configuring TCP-UDP Proxy Servers

The TCP/UDP Proxy Servers table lets you configure up to 10 TCP/UDP proxy servers. This table allows you to configure the device as a proxy for other applications that are not based on HTTP. For example, it can be used to intermediate between clients and a DNS server for DNS lookup or between clients and an NTP server for clock synchronization.

The following procedure describes how to configure a TCP-UDP Proxy Server through the Web interface. You can also configure it through ini file [TcpUdpServer] or CLI (`configure network > http-proxy > tcp-udp-server`).

➢ To configure a TCP/UDP Proxy Server:

1. Enable the HTTP Proxy application, as described in Enabling the HTTP Proxy Application on page 342.
2. Open the TCP/UDP Proxy Servers table (**Setup** menu > **IP Network** tab > **HTTP Proxy** folder > **TCP/UDP Proxy Servers**).
3. Click **New**; the following dialog box appears:



4. Configure a TCP/UDP Proxy Server according to the parameters described in the table below.
5. Click **Apply**, and then save your settings to flash memory.

Table 16-22:TCP/UDP Proxy Servers Table Parameter Descriptions

Parameter	Description
General	
'Index'	Defines an index number for the new table row.

Parameter	Description
[TcpUdpServer_Index]	**Note:** ■ Each row must be configured with a unique index. ■ The parameter is mandatory.
'Name' `name` [TcpUdpServer_Name]	Defines a descriptive name, which is used when associating the row in other tables. The valid value is a string of up to 40 characters. By default, no value is defined. **Note:** ■ Each row must be configured with a unique name. ■ The parameter is mandatory.
'Additional Directive Set' `directive-set` [TcpUdpServer_ AdditionalDirectiveSet]	Assigns an NGINX Directive Set for the HTTP service. To configure HTTP Directive Sets, see Configuring HTTP Directive Sets on page 360.
Listen Parameters	
'Listening Interface' `listen-interface` [TcpUdpServer_ ListeningInterface]	Assigns a local IP network interface for the listening (source) interface for communication with the TCP-UDP proxy server. To configure IP Interfaces, see Configuring IP Network Interfaces on page 124. By default, no value is defined. **Note:** ■ The parameter is mandatory. ■ The NGINX directive for this parameter is "listen ip".
'TCP Listening Port' `tcp-port` [TcpUdpServer_ TCPListeningPort]	Defines the TCP port of the listening interface. **Note:** ■ You must configure a TCP and/or UDP port. ■ The NGINX directive for this parameter is "listen ip:port". ■ The source ports used for outgoing TCP connections are not configurable and are dynamically determined by the device in the range of 32,768-61,000.
'UDP Listening Port' `udp-port`	Defines the TCP port of the listening interface. **Note:**

Parameter	Description
[TcpUdpServer_ UDPListeningPort]	■ You must configure a TCP and/or UDP port. ■ The NGINX directive for this parameter is "listen ip:port udp".
'Listen Side SSL' `listen-use-ssl` [TcpUdpServer_ ListenUseSSL]	Enables TLS on the listening side (i.e., listening to incoming connection requests). ■ [0] **Disable** (default) ■ [1] **Enable** **Note:** The NGINX directive for this parameter is "listen ip:port ssl".
'Listen TLS Context' `listen-tls-context` [TcpUdpServer_ ListenTLSContext]	Assigns a TLS Context (TLS certificate) for the listening side. This is required if you have configured the 'Listen Side SSL' parameter to **Enable** (see above). To configure TLS Contexts, see Configuring TLS Certificate Contexts on page 158. **Note:** The NGINX directives for this parameter is "ssl_ certificate", "ssl_certificate_key", "ssl_ciphers", "ssl_ protocols", and "ssl_password_file".
Upstream Parameters	
'Upstream Group' `upstream-group` [TcpUdpServer_ UpstreamGroup]	Assigns a group of servers (Upstream Group) to which to forward connection requests. To configure Upstream Groups, see Configuring Upstream Groups on page 356. **Note:** ■ Only Upstream Groups with TCP/UDP interfaces can be assigned. ■ The NGINX directive for this parameter is "proxy_pass upstream".
'Outbound Interface' `outbound-interface` [TcpUdpServer_ OutboundInterface]	Assigns a local, IP network interface for communicating with the Upstream Group. To configure IP network interfaces, see Configuring IP Network Interfaces on page 124. By default, no value is defined. **Note:** ■ The parameter is mandatory. ■ The NGINX directive for this parameter is "proxy_ bind".

Parameter	Description
'Upstream Side SSL' `upstream-use-ssl` [TcpUdpServer_ UpstreamUseSSL]	Enables TLS for securing connection requests with the Upstream Group. ■ [0] **Disable** (default) ■ [1] **Enable** **Note:** ■ If configured to **Enable**, you must assign a TLS Context (see the 'Upstream TLS Context' parameter below). ■ The NGINX directive for this parameter is "proxy_ssl on".
'Upstream TLS Context' `upstream-tls-context` [TcpUdpServer_ UpstreamTLSContext]	Assigns a TLS Context for the TLS connection with the HTTP location. To configure TLS Contexts, see Configuring TLS Certificate Contexts on page 158. **Note:** ■ The parameter is applicable only if the 'Upstream Side SSL' parameter is configured to **Enable** (see above). ■ The NGINX directives for this parameter are "proxy_ ssl_certificate", "proxy_ssl_certificate_key", "proxy_ ssl_ciphers", "proxy_ssl_protocols", and "proxy_ssl_ password_file".
'Upstream Verify Certificate' `upstream-verify-cert` [TcpUdpServer_ UpstreamVerifyCertificate]	Enables TLS certificate verification of the Upstream Host on outgoing connection requests to the Upstream Group, when the connection is TLS. ■ [0] **No** = (Default) No certificate verification is done. ■ [1] **Yes** = The device verifies the authentication of the certificate received from the host. The device authenticates the certificate against the trusted root certificate store associated with the assigned TLS Context (see 'Upstream TLS Context' parameter above) and if ok, allows communication with the host. If authentication fails, the device denies communication (i.e., handshake fails). The device can also authenticate the certificate by querying with an Online Certificate Status Protocol (OCSP) server whether the certificate has been revoked. This is also configured for the associated TLS Context. **Note:**

Parameter	Description
	■ The parameter is applicable only if the 'Upstream Side SSL' parameter is configured to **Enable** (see above). ■ The NGINX directive for this parameter is "proxy_ssl_verify".

Configuring Upstream Groups

The Upstream Groups table lets you configure up to 10 Upstream Groups. Once configured, you can configure Upstream Hosts for the Upstream Group (see Configuring Upstream Hosts on page 358).

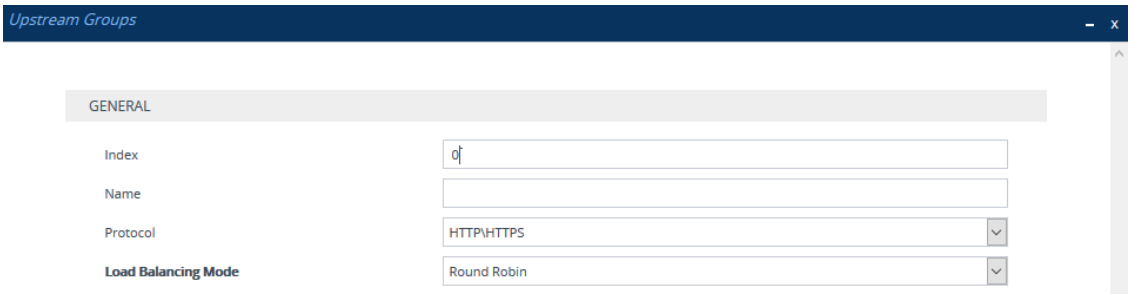
An Upstream Group is a set of one or more hosts (*Upstream Host*) that can serve a particular set of data. The HTTP Proxy distributes the requests among the members (hosts) of the Upstream Group according to the specified load balancing mode.

The Upstream Group may be made up of one or more primary hosts and zero or more backup hosts. HTTP requests for the Upstream Group are distributed among all the primary hosts. Backup hosts do not receive requests unless all the primary hosts are down.

The following procedure describes how to configure Upstream Groups through the Web interface. You can also configure it through ini file [UpstreamGroup] or CLI (`configure network > http-proxy > upstream-group`).

➢ To configure an Upstream Group:

1. Enable the HTTP Proxy application, as described in Enabling the HTTP Proxy Application on page 342.
2. Open the Upstream Groups table (**Setup** menu > **IP Network** tab > **HTTP Proxy** folder > **Upstream Groups**).
3. Click **New**; the following dialog box appears:



4. Configure an Upstream Group according to the parameters described in the table below.
5. Click **Apply**, and then save your settings to flash memory.

Table 16-23:Upstream Groups Table Parameter Descriptions

Parameter	Description
General	
'Index' [UpstreamGroup_Index]	Defines an index number for the new table row. **Note:** ■ Each row must be configured with a unique index. ■ The parameter is mandatory.
'Name' `name` [UpstreamGroup_Name]	Defines a descriptive name, which is used when associating the row in other tables. The valid value is a string of up to 40 characters. By default, no value is defined. **Note:** ■ Each row must be configured with a unique name. ■ The NGINX directive for this parameter is "upstream name { … }". ■ The parameter is mandatory. ■ The parameter value cannot contain a forward slash (/).
'Protocol' `protocol` [UpstreamGroup_Protocol]	Defines the protocol. ■ [0] **HTTP/HTTPS** (default) ■ [1] **TCP/UDP** **Note:** ■ To assign the Upstream Group to a TCP/UDP Proxy Server, configure the parameter to TCP/UDP. To configure TCP/UDP Proxy Servers, see Configuring TCP-UDP Proxy Servers on page 352. ■ To assign the Upstream Group to an HTTP Proxy Server, configure the parameter to HTTP/HTTPS. To configure HTTP Proxy Servers, see Configuring HTTP Proxy Servers on page 343. ■ For NGINX, the parameter determines nesting within the "http" or "stream" context.
'Load Balancing Mode' `load-balancing-mode`	Defines the load-balancing of traffic method for the hosts belonging to the Upstream Group. ■ [0] **Round Robin** = (Default) Traffic requests are balanced

Parameter	Description
[UpstreamGroup_ LoadBalancingMode]	across all hosts. Every consecutive request is sent to the next available host. ■ [1] **IP Hash** = All requests from a given client (by IP address) is sent to the same host, regardless of current load. ■ [2] **Least Connections** = New requests are sent to the host with the fewest active connections. **Note:** The NGINX directive for this parameter is "ip-hash (1)", "least-conn (2)", and "round-robin (0)".

Configuring Upstream Hosts

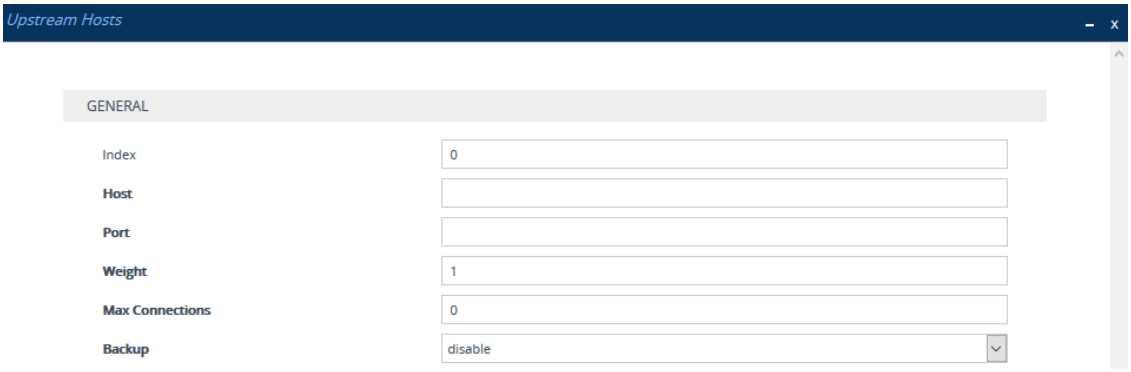
The Upstream Hosts table lets you configure up to 50 Upstream Hosts. The Upstream Hosts table is a "child" of the Upstream Group table (see Configuring Upstream Groups on page 356) and therefore, Upstream Hosts are configured per Upstream Group. Up to 5 Upstream Hosts can be configured per Upstream Group.

The following procedure describes how to configure Upstream Hosts through the Web interface. You can also configure it through ini file [UpstreamHost] or CLI (`configure network > http-proxy > upstream-host`).

The device activates a keep-alive mechanism to ensure that the hosts configured in the Upstream Hosts table are accessible through HTTP. If an HTTP server goes offline, the device raises an alarm. The device also notifies this event in Syslog messages if you have configured the 'HTTP Proxy Debug Level' parameter to **Info** or **Notice**.

➢ **To configure an Upstream Host:**

1. Enable the HTTP Proxy application, as described in Enabling the HTTP Proxy Application on page 342.
2. Open the Upstream Groups table (**Setup** menu > **IP Network** tab > **HTTP Proxy** folder > **Upstream Groups**).
3. In the table, select the required Upstream Group index row, and then click the **Upstream Hosts** link located below the table; the Upstream Hosts table appears.
4. Click **New**; the following dialog box appears:



5. Configure an Upstream Host according to the parameters described in the table below.
6. Click **Apply**, and then save your settings to flash memory.

Table 16-24:Upstream Hosts Table Parameter Descriptions

Parameter	Description
'Index' [UpstreamHost_HostIndex]	Defines an index number for the new table row. **Note:** ■ Each row must be configured with a unique index. ■ The parameter is mandatory.
'Host' `host` [UpstreamHost_Host]	Defines the address of the host as an FQDN or IP address (in dotted-decimal notation). **Note:** The NGINX directive for this parameter is "server host:port".
'Port' `port` [UpstreamHost_Port]	Defines the port number. **Note:** ■ If the Upstream Group to which the host belongs is configured to use the TCP/UDP protocol, then this parameter must be configured. ■ The NGINX directive for this parameter is "server ip:port". If not configured, NGINX defaults to 80 for HTTP and 443 for HTTPS.
'Weight' `weight` [UpstreamHost_Weight]	Defines the weight for the load balancer. The load balancer distributes the requests among the hosts in the Upstream Group based on the weight of each host. For example, if host A is configured with a weight of 3, host B with a weight of 1, and host C with a weight of 1, then in each cycle, the load balancer will send three requests to host A, one

Parameter	Description
	request to host B and one request to host C. The valid range is 1 to 100. The default is 1 (i.e., each host in the Upstream Group has equal weight). **Note:** The NGINX directive for this parameter is "server ip:port weight=n".
'Max Connections' `max-connections` [UpstreamHost_MaxConnections]	Defines the maximum number of simultaneous active connections to the proxied Upstream Host. The default is 0 (i.e., unlimited). **Note:** The NGINX directive for this parameter is "server ip:port max_conns=n".
'Backup' `backup` [UpstreamHost_Backup]	Enables the host to serve as a backup host. The backup host does not receive any requests unless all the primary hosts in the Upstream Group are down. ■ [0] **Disable** (default) ■ [1] **Enable** **Note:** The NGINX directive for this parameter is "server ip:port backup".

Configuring HTTP Directive Sets

The HTTP Directive Sets table lets you configure up to 30 HTTP Directive Sets. The table lets you configure additional custom directives to HTTP Proxy server configuration. These directives are reflected in the configuration file generated for the NGINX HTTP Proxy. The directives of each HTTP Directive Set is configured in the HTTP Directives table (see Configuring HTTP Directives on page 362), which is a "child" of the HTTP Directive Sets table.

Directives are grouped into Directive Sets, which you can then assign to HTTP Proxy Servers (see Configuring HTTP Proxy Servers on page 343), HTTP Locations (see Configuring HTTP Locations on page 348), and TCP/UDP Proxy Servers (see Configuring TCP-UDP Proxy Servers on page 352), using the 'Additional Directive Set' parameter in their respective tables.

For example, to control behavior of specific encoding and communication parameters relating to a particular location, you can configure the following NGINX directives:

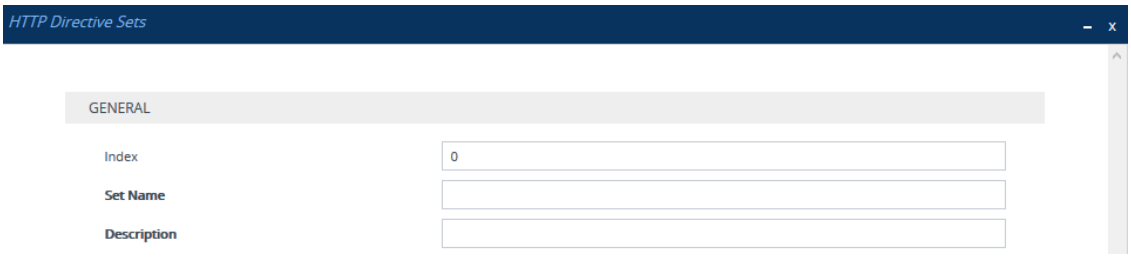
```
chunked_transfer_encoding off;
keepalive_timeout 50s;
```

- The device does not validate Directive Sets, which it passes directly to the NGINX configuration file. If the configured directives are not entered using the correct syntax, NGINX rejects the new configuration. For more information, refer to the NGINX documentation at http://nginx.org/en/docs . An alphabetical index to all directives can be found at http://nginx.org/en/docs/dirindex.html.
- By default, the device is configured with an HTTP Directive Set for rate limiting. This directive ensures that priority is given to network traffic carrying SIP signaling and media over HTTP traffic. It is highly recommended to configure these limitations on the HTTP Proxy. This HTTP Directive Set includes the following directives:
  - ✓ "limit_conn": Specifies the maximum number of simultaneous client connections (default 100).
  - ✓ "limit_rate": Specifies the bandwidth limit per connection (bytes per second). This syntax supports a suffix of "k" for kilobytes and "m" for megabytes. The default is 0.

The following procedure describes how to configure HTTP Directive Sets through the Web interface. You can also configure it through ini file [HTTPDirectiveSets] or CLI (`configure network > http-proxy > directive-sets`).

➢ To configure an HTTP Directive Set:

1. Enable the HTTP Proxy application, as described in Enabling the HTTP Proxy Application on page 342.
2. Open the HTTP Directive Sets table (**Setup** menu > **IP Network** tab > **HTTP Proxy** folder > **HTTP Directive Sets**).
3. Click **New**; the following dialog box appears:



4. Configure an HTTP Directive Set according to the parameters described in the table below.
5. Click **Apply**, and then save your settings to flash memory.
6. Configure directives for the HTTP Directive Set (see Configuring HTTP Directives on the next page).

Table 16-25:HTTP Directive Sets Table Parameter Descriptions

Parameter	Description
'Index'	Defines an index number for the new table row.

Parameter	Description
[HTTPDirectiveSets_ Index]	**Note:** ■ Each row must be configured with a unique index. ■ The parameter is mandatory.
'Set Name' `set-name` [HTTPDirectiveSets_ SetName]	Defines a descriptive name, which is used when associating the row in other tables. The valid value is a string of up to 40 characters. By default, no value is defined. **Note:** The parameter value cannot contain a forward slash (/).
'Description' `set-description` [HTTPDirectiveSets_ Description]	Defines a brief description for the HTTP Directive Set. The valid value is a string of up to 100 characters. By default, no value is defined.

Configuring HTTP Directives

The HTTP Directives table lets you configure up to 500 HTTP Directives. The table is a "child" of the HTTP Directive Sets table (see Configuring HTTP Directive Sets on page 360).

> When generating the NGINX configuration file, the device includes the resolver directive, specifying the primary and secondary DNS servers, as configured in Configuring a DNS Server for HTTP Services on page 343. However, NGINX supports optional parameters that allow you to fine-tune the behavior of the DNS resolution. You can include these additional parameters using the ini file parameter [NginxResolverParams), which is added to the resolver directive when the device generates the NGINX configuration file. For more information on these optional parameters, go to the NGINX forum.

The following procedure describes how to configure HTTP Directives through the Web interface. You can also configure it through ini file [HTTPDirectives] or CLI (`configure network > http-proxy > directives`).

➢ To configure an HTTP Directive:

1. Enable the HTTP Proxy application, as described in Enabling the HTTP Proxy Application on page 342.
2. Open the HTTP Directive Sets table (**Setup** menu > **IP Network** tab > **HTTP Proxy** folder > **HTTP Directive Sets**).
3. In the table, select the required HTTP Directive Set index row, and then click the **HTTP Directives** link located below the table; the HTTP Directives table appears.
4. Click **New**; the following dialog box appears:

5. Configure an HTTP Directive according to the parameters described in the table below.
6. Click **Apply**, and then save your settings to flash memory.

Table 16-26:HTTP Directives Table Parameter Descriptions

Parameter	Description
'Index' [HTTPDirectives_ RowIndex]	Defines an index number for the new table row. **Note:** ■ Each row must be configured with a unique index. ■ The parameter is mandatory.
' Directive' `directive` [HTTPDirectives_ Directive]	Defines an NGINX directive. **Note:** ■ The parameter is mandatory. ■ Make sure that you end the directive with a semicolon (;).

Configuring an HTTP-based OVOC Service

The OVOC Services table lets you configure a single HTTP-based AudioCodes One Voice Operations Center (OVOC) service. You can configure the device to act as an HTTP Proxy that enables OVOC to manage AudioCodes equipment (such as IP Phones) over HTTP when the equipment is located behind NAT (e.g., in the LAN) and OVOC is located in a public domain (e.g., in the WAN). This setup resolves NAT traversal issues. The IP Phones register with the device to allow communication between the IP Phones and OVOC. Once setup, the OVOC administrator can access the Web-based management interfaces of each IP Phone .

A summary of the steps required to configure an HTTP Proxy for this OVOC service is listed below:

1. Enable the HTTP Proxy application (see Enabling the HTTP Proxy Application).
2. Configure two local, listening IP network interfaces - one for OVOC and one for the IP Phones (see Configuring IP Network Interfaces on page 124).
3. Configure the OVOC service in the OVOC Services table (described below). This entails specifying the IP network interfaces as well as the port number within each interface to which the HTTP Proxy must listen to.

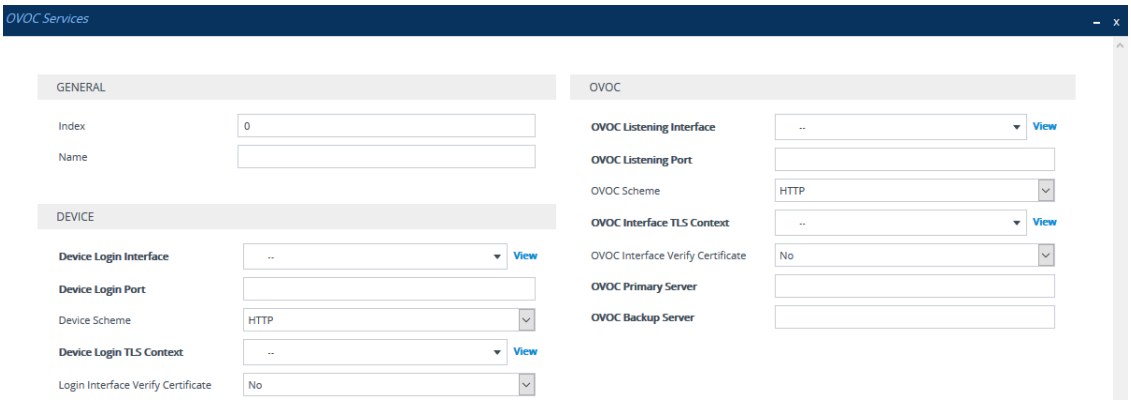
4. Configure the device's firewall (Firewall table) to allow incoming traffic from OVOC. For more information, see Configuring Firewall Rules to Allow Incoming OVOC Traffic on page 184.

- It is recommended **not** to use port 80 as this is the default port used by IP Phones for their Web-based management interface.
- No special configuration is required on the managed equipment.

The following procedure describes how to configure an OVOC service through the Web interface. You can also configure it through ini file [OVOCService] or CLI (`configure network > http-proxy > ovoc-serv`).

➢ To configure an OVOC Service:

1. Enable the HTTP Proxy application, as described in Enabling the HTTP Proxy Application on page 342.
2. Open the OVOC Services table (**Setup** menu > **IP Network** tab > **HTTP Proxy** folder > **OVOC Services**).
3. Click **New**; the following dialog box appears:



4. Configure an OVOC Service according to the parameters described in the table below.
5. Click **Apply**, and then save your settings to flash memory.

Table 16-27:OVOC Services Table Parameter Descriptions

Parameter	Description
General	
'Index' [OVOCService_Index]	Defines an index number for the new table row. **Note:** ■ Each row must be configured with a unique index. ■ The parameter is mandatory.

Parameter	Description
'Name' service-name [OVOCService_ServiceName]	Defines a descriptive name, which is used when associating the row in other tables. The valid value is a string of up to 40 characters. By default, no value is defined. **Note:** ■ Each row must be configured with a unique name. ■ The parameter is mandatory.
Device	
'Device Login Interface' device-login-interface [OVOCService_ DeviceLoginInterface]	Assigns an IP network interface (local, listening HTTP interface:port) for communication with the client. To configure IP Interfaces, see Configuring IP Network Interfaces on page 124. By default, no value is defined. **Note:** ■ The parameter is mandatory. ■ The NGINX directive for this parameter is "proxy_ bind".
'Device Login Port' device-login-port [OVOCService_DeviceLoginPort]	Defines the login port of the requesting client. **Note:** The NGINX directive for this parameter is "proxy_bind".
'Device Scheme' device-scheme [OVOCService_DeviceScheme]	Defines the protocol for communication with the requesting client. ■ [0] **HTTP** (default) ■ [1] **HTTPS** **Note:** If configured to **HTTPS**, you must assign a TLS Context (see the 'Device Login TLS Context' parameter, below).
'Device Login TLS Context' device-login-tls-context [OVOCService_ LoginInterfaceTLSContext]	Assigns a TLS Context (TLS configuration) for the interface with the requesting client. This is required if you have configured the 'Device Scheme' parameter to **HTTPS** (see above). To configure TLS Contexts, see Configuring TLS Certificate Contexts on page 158. **Note:** The NGINX directive for this parameter is

Parameter	Description
	"proxy_ssl_certificate", "proxy_ssl_certificate_key", "proxy_ssl_ciphers", and "proxy_ssl_protocols".
'Device Login Interface Verify Certificate' `device-interface-verify-cert` [OVOCService_ LoginInterfaceVerifyCert]	Enables the verification of the TLS certificate that is used in the incoming client connection request. ■ [0] **No** = (Default) No certificate verification is done. ■ [1] **Yes** = The device verifies the authentication of the certificate received from the client. The device authenticates the certificate against the trusted root certificate store associated with the assigned TLS Context (see 'Device Login TLS Context' parameter above) and if ok, allows communication with the client. If authentication fails, the device denies communication (i.e., handshake fails). The device can also authenticate the certificate by querying with an Online Certificate Status Protocol (OCSP) server whether the certificate has been revoked. This is also configured for the associated TLS Context. **Note:** The NGINX directive for this parameter is "proxy_ssl_verify".
OVOC	
'OVOC Listening Interface' `ovoc-interface` [OVOCService_ OVOCListeningInterface]	Assigns an IP network interface (local, listening HTTP interface:port) for communication with OVOC. To configure IP Interfaces, see Configuring IP Network Interfaces on page 124. By default, no value is defined. **Note:** ■ The parameter is mandatory. ■ The NGINX directive for this parameter is "proxy_ bind".
'OVOC Listening Port' `ovoc-port` [OVOCService_OVOCListeningPort]	Defines the listening port for the OVOC interface. **Note:** The NGINX directive for this parameter is "proxy_bind".
'OVOC Scheme' `ovoc-scheme`	Defines the security scheme for the connection with OVOC.

Parameter	Description
[OVOCService_OVOCScheme]	■ [0] **HTTP** (default) ■ [1] **HTTPS** **Note:** ■ If configured to **HTTPS**, you must assign a TLS Context (see the 'OVOC Interface TLS Context' parameter, below). ■ The NGINX directive for this parameter is "proxy_pass scheme://upstream".
'OVOC Interface TLS Context' `ovoc-interface-tls-context` [OVOCService_OVOCInterfaceTLSContext]	Assigns a TLS Context (TLS configuration) for the OVOC listening interface. This is required if you have configured the 'OVOC Scheme' parameter to **HTTPS** (see above). To configure TLS Contexts, see Configuring TLS Certificate Contexts on page 158. **Note:** The NGINX directive for this parameter is "proxy_ssl_certificate", "proxy_ssl_certificate_key", "proxy_ssl_ciphers", and "proxy_ssl_protocols".
'OVOC Interface Verify Certificate' `ovoc-verify-cer` [OVOCService_OVOCInterfaceVerifyCert]	Enables the verification of the TLS certificate that is used in the incoming connection request from OVOC. ■ [0] **No** = (Default) No certificate verification is done. ■ [1] **Yes** = The device verifies the authentication of the certificate received from OVOC. The device authenticates the certificate against the trusted root certificate store associated with the assigned TLS Context (see 'OVOC Interface TLS Context' parameter above) and if ok, allows communication with OVOC. If authentication fails, the device denies communication (i.e., handshake fails). The device can also authenticate the certificate by querying with an Online Certificate Status Protocol (OCSP) server whether the certificate has been revoked. This is also configured for the associated TLS Context. **Note:** The NGINX directive for this parameter is "proxy_ssl_verify".
'OVOC Primary Server'	Defines the address of the primary OVOC server.

Parameter	Description
`primary-server` [OVOCService_PrimaryServer]	**Note:** ■ This parameter is mandatory. ■ When you configure this parameter, an Upstream Group is automatically added (see Configuring Upstream Groups on page 356). ■ The NGINX directive for this parameter is "upstream ems { addr1, addr2 backup }" and "proxy_pass scheme://ems".
'OVOC Backup Server' `backup-server` [OVOCService_BackupServer]	Defines the address of the secondary OVOC server. **Note:** ■ When you configure this parameter, an Upstream Group is automatically added. ■ The NGINX directive for this parameter is "upstream ems { addr1, addr2 backup }" and "proxy_pass scheme://ems".

Troubleshooting NGINX Configuration

Troubleshooting may be necessary when configuring your HTTP or TCP/UDP proxy services with NGINX directives. Due to the large and complex dictionary of directives supported by NGINX and their complex grammatical structure, the device assists you by validating your configured directives. It does this only once you have applied them (i.e., clicked the **Apply** button) in the HTTP Directives table (see Configuring HTTP Directives on page 362).

In addition, the device generates the following NGINX configuration files:

- **nginx.conf:** This file contains the currently active configuration, which is valid.
- **temp_nginx.conf:** This file is generated if you have invalid configuration (directive errors). It is a temporary file and contains your new configuration, which is invalid. It is applied only if the device is restarted.
- **nginx.errors:** This file is generated if you have invalid configuration (directive errors). This file contains all the error messages, indicating the line on which the error exists in the temp_nginx.conf file.

If you have modified your configuration and errors occur, the device continues running with the previous, valid NGINX configuration, unless the device is restarted, in which case it applies and uses the modified configuration.

In addition, if an NGINX validation error exists during configuration or if the device restarts with an invalid NGINX configuration, the device indicates this by the following:

- Sends an alarm to the Active Alarms table ("NGINX configuration file is not valid")

- Sends the error to Syslog, which is marked with "http_app"

To send the NGINX files to a remote destination in tar fiel format (.tar), use the following CLI command:

```
# copy nginx-conf-files to <Protocol>://<Address>/<filename>.tar
```

To view the NGINX files in CLI, use the following command:

```
show network http-proxy conf active|errors|new
```

Configuring a Public IP Address for NGINX NAT Traversal

When the device is located behind NAT, OVOC can only communicate with the device's embedded NGINX HTTP-based proxy using the device's public static NAT address. However, by default, the device sends OVOC its' private address. The device's address (private or public) appears in the proprietary X-AC-Proxy-URL header in HTTP requests that the device sends to OVOC.

➢ **To configure a public IP address for HTTP Proxy:**

1. Open the General Settings page (**Setup** menu > **IP Network** tab > **HTTP Proxy** folder > **General Settings**).

2. In the 'HTTP Proxy Global Address' [HttpProxyGlobalAddress] field, enter the public IP address.
3. Click **Apply**.

E9-1-1 Support for Microsoft Teams and Skype for Business

The Enhanced 9-1-1 (E9-1-1) service is becoming the mandatory emergency service required in many countries around the world. The E9-1-1 service, based on its predecessor 911, enables emergency operators to pinpoint the location (granular location) of callers who dial the 9-1-1 emergency telephone number.

Today, most companies implement an IP-based infrastructure providing a VoIP network with fixed and nomadic users, allowing connectivity anywhere with any device. This, together with an often deployed multi-line telephone system (MLTS) poses a challenge for E9-1-1 due to the difficulty in accurately locating the E9-1-1 caller.

This section describes the E9-1-1 solution provided by Microsoft Teams / Skype for Business and AudioCodes' device's ELIN interworking capabilities, which provides the SIP Trunk connectivity to the E9-1-1 emergency service provider. This section also describes the configuration of the

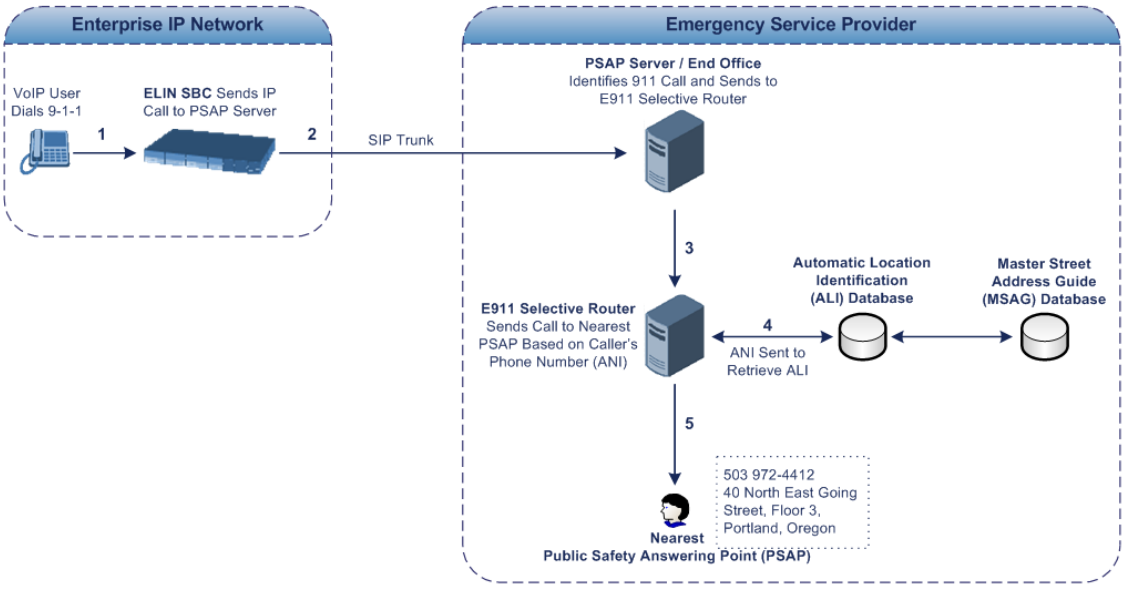
device for interoperating between the Teams / Skype for Business environment and the E9-1-1 emergency provider.

- The ELIN feature for E9-1-1 is a license-based feature and is available only if it is included in the License Key installed on the device. For ordering the feature, please contact the sales representative of your purchased device. For installing a new License Key, see License Key.

About E9-1-1 Services

E9-1-1 is a national emergency service for many countries, enabling E9-1-1 operators to automatically identify the geographical location and phone number of a 911 caller. In E9-1-1, the 911 caller is routed to the nearest E9-1-1 operator, termed *public safety answering point* (PSAP) based on the location of the caller. Automatic identification of the caller's location and phone number reduces the time spent on requesting this information from the 911 caller. Therefore, the E9-1-1 service enables the PSAP to quickly dispatch the relevant emergency services (for example, fire department or police) to the caller's location. Even if the call prematurely disconnects, the operator has sufficient information to call back the 911 caller.

The figure below illustrates the routing of an E9-1-1 call to the PSAP:



1. The VoIP user dials 9-1-1.
2. The AudioCodes' ELIN device sends the call to the emergency service provider over the SIP Trunk (PSAP server).
3. The emergency service provider identifies the call is an emergency call and sends it to an E9-1-1 Selective Router in the Emergency Services provider's network.
4. The E9-1-1 Selective Router determines the geographical location of the caller by requesting this information from an Automatic Location Identification (ALI) database based on the phone number or Automatic Number Identifier (ANI) of the 911 caller. Exact

location information is also supplied by the Master Street Address Guide (MSAG) database, which is a companion database to the ALI database. Phone companies and public safety agencies collaborate beforehand to create master maps that match phone numbers, addresses and cross streets to their corresponding PSAP. This MSAG is the official record of valid streets (with exact spelling), street number ranges, and other address elements with which the service providers are required to update their ALI databases.

5. The E9-1-1 Selective Router sends the call to the appropriate PSAP based on the retrieved location information from the ALI.
6. The PSAP operator dispatches the relevant emergency services to the E9-1-1 caller.

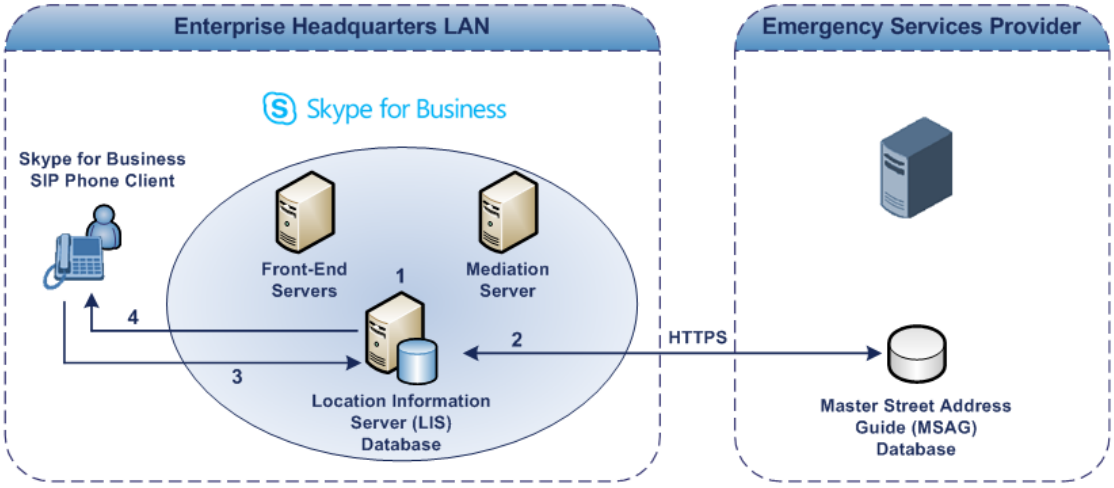
Microsoft Skype for Business and E9-1-1

Microsoft Skype for Business enables Enterprise voice users to access its unified communications platform from virtually anywhere and through many different devices. This, together with a deployed MLTS, poses a challenge for E9-1-1 due to the difficulty in accurately locating the E9-1-1 caller. However, Skype for Business offers an innovative solution to solving Enterprises E9-1-1 location problems.

Gathering Location Information of Skype for Business Clients for 911 Calls

When a Microsoft Skype for Business client is enabled for E9-1-1, the location data that is stored on the client is sent during an emergency call. This stored location information is acquired automatically from the Microsoft Location Information Server (LIS). The LIS stores the location of each network element in the enterprise. Immediately after the Skype for Business client registration process or when the operating system detects a network connection change, each Skype for Business client submits a request to the LIS for a location. If the LIS is able to resolve a location address for the client request, it returns the address in a location response. Each client then caches this information. When the Skype for Business client dials 9-1-1, this location information is then included as part of the emergency call and used by the emergency service provider to route the call to the correct PSAP.

The gathering of location information in the Skype for Business network is illustrated in the figure below:



1. The Administrator provisions the LIS database with the location of each network element in the Enterprise. The location is a civic address, which can include contextual in-building and company information. In other words, it associates a specific network entity (for example, a WAP) with a physical location in the Enterprise (for example, Floor 2, Wing A, and the Enterprise's street address). For more information on populating the LIS database, see Adding ELINs to the Location Information Server.
2. The Administrator validates addresses with the emergency service provider's MSAG –a companion database to the ALI database. This ensures that the civic address is valid as an official address (e.g., correct address spelling).
3. The Skype for Business client initiates a location request to the LIS under the following circumstances:
   - Immediately after startup and registering the user with Skype for Business
   - Approximately every four hours after initial registration
   - Whenever a network connection change is detected (such as roaming to a new WAP)

   The Skype for Business client includes in its location request the following known network connectivity information:
   - Always included:
     - IPv4 subnet
     - Media Access Control (MAC) address
   - Depends on network connectivity:
     - Wireless access point (WAP) Basic Service Set Identifier (BSSID)
     - Link Layer Discovery Protocol-Media Endpoint Discovery (LLDP-MED) chassis ID and port ID

   For a Skype for Business client that moves inside the corporate network such as a soft phone on a laptop that connects wirelessly to the corporate network, Skype for Business can determine which subnet the phone belongs to or which WAP / SSID is currently serving the soft-client.
4. The LIS queries the published locations for a location and if a match is found, returns the location information to the client. The matching order is as follows:
   - WAP BSSID
   - LLDP switch / port
   - LLDP switch
   - Subnet
   - MAC address

   This logic ensures that for any client that is connected by a wireless connection, a match is first attempted based on the hardware address of its connected access point. The logic is for the match to be based on the most detailed location. The subnet generally provides the

least detail. If no match is found in the LIS for WAP BSSID, LLDP switch / port, LLDP switch, or subnet, the LIS proxies the MAC address to an integrated Simple Network Management Protocol (SNMP) scanning application. Using SNMP may benefit some organizations for the following reasons:

- LLDP is not supported by Skype for Business so this provides a mechanism for soft phones to acquire detailed location information.
- Installed Layer-2 switches may not support LLDP.

If there is no match and the LIS cannot determine the location, the user may be prompted to manually enter the location. For example, the client may be located in an undefined subnet, at home, in a coffee shop or anywhere else outside the network. When a user manually provides a location, the location is mapped based on the MAC address of the default gateway of the client's network and stored on the client. When the client returns to any previously stored location, the client is automatically set to that location. A user can also manually select any location stored in the local users table and manage existing entries.

Adding ELINs to the Location Information Server

As mentioned in the previous section, the administrator needs to populate the Location Information Server (LIS) database with a network wire map, which maps the company's network elements to civic addresses. Once done, it can automatically locate clients within a network. You can add addresses individually to the LIS or in a batch using a comma-separated value (CSV) file containing the column formats for the network elements, as listed below:

- **Wireless access point:**
  <BSSID>,<Description>,<Location>,<**CompanyName**>,<HouseNumber>, <HouseNumberSuffix>,<PreDirectional>,<StreetName>,<StreetSuffix>,<PostDirectional>, <City>,<State>, <PostalCode>,<Country>
- **Subnet:** <Subnet>,<Description>,<Location>,<CompanyName>,<HouseNumber>, <HouseNumberSuffix>,<PreDirectional>,<StreetName>, <StreetSuffix>,<PostDirectional>, <City>,<State>,<PostalCode>,<Country>
- **Port:** <ChassisID>,<PortIDSubType>,<PortID>,<Description>,<Location>,<CompanyName>, <HouseNumber>, <HouseNumberSuffix>,<PreDirectional>,<StreetName>, <StreetSuffix>,<PostDirectional>,<City>,<State>,<PostalCode>,<Country>
- **Switch:** <ChassisID>,<Description>,<Location>,<CompanyName>, <HouseNumber>,<HouseNumberSuffix>,<PreDirectional>,<StreetName>,<StreetSuffix>, <PostDirectional>, <City>,<State>,<PostalCode>,<Country>

For the ELIN number to be included in the SIP INVITE (XML-based PIDF-LO message) sent by the Mediation Server to the ELIN device, the administrator must add the ELIN number to the <CompanyName> column (shown in the table above in bold typeface). As the ELIN device supports up to five ELINs per PIDF-LO, the <CompanyName> column can be populated with up to this number of ELINs, each separated by a semicolon. The digits of each ELIN can be separated by hyphens (xxx-xxx-xxx) or they can be adjacent (xxxxxxxxx). When the ELIN device receives the SIP INVITE message, it extracts the ELINs from the ELIN field in the PIDF-LO (e.g.,

<ca:ELIN>1111-222-333; 1234567890 </ca:ELIN>), which corresponds to the <CompanyName> column of the LIS.

> For backward compatibility, if the ELIN field doesn't appear in the PIDF-LO, the device extracts the ELINs from the NAM field.

If you do not populate the location database and the Skype for Business location policy, and Location Required is set to **Yes** or **Disclaimer**, the user is prompted to enter a location manually.

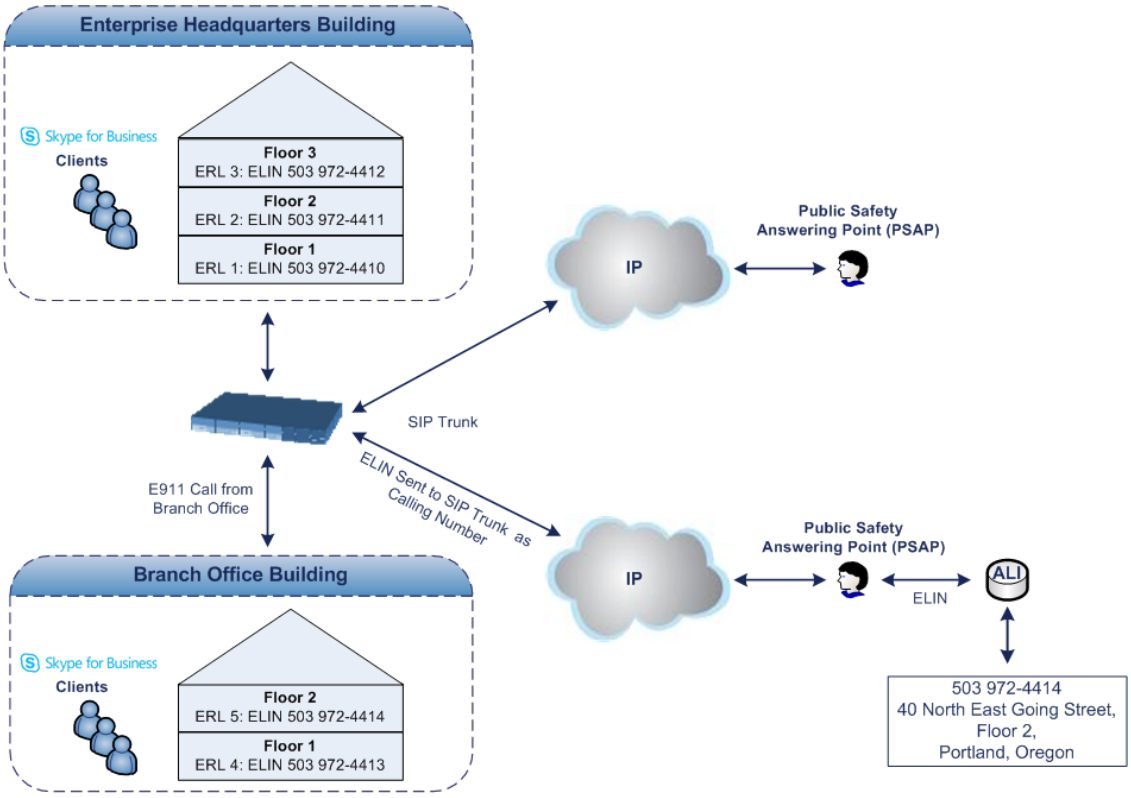
Passing Location Information to the PSTN Emergency Provider

When a Skype for Business client, enabled for E9-1-1 emergency services, dials 9-1-1, the location data and callback information stored on the client is sent with the call through the Mediation Server to a SIP Trunk-based emergency service provider. The emergency service provider then routes the call to the nearest and most appropriate PSAP based on the location information contained within the call.

Skype for Business passes the location information of the Skype for Business client in an IETF-standard format - Presence Information Data Format - Location Object (PIDF-LO)–in a SIP INVITE message. However, this content cannot be sent on the SIP Trunksince they do no support such a content. To overcome this, Enterprises deploying the device can divide their office space into Emergency Response Locations (ERLs) and assign a dedicated Emergency Location Identification Number (ELIN) to each ERL (or zone). When Skype for Business sends a SIP INVITE message with the PIDF-LO to the device, it can parse the content and translate the calling number to an appropriate ELIN. The device then sends the call to the SIP Trunkwith the ELIN number as the calling number. The ELIN number is sent to the emergency service provider, which sends it on to the appropriate PSAP according to the ELIN address match in the ALI database lookup.

The ERL defines a specific location at a street address, for example, the floor number of the building at that address. The geographical size of an ERL is according to local or national regulations (for example, less than 7000 square feet per ERL). Typically, you would have an ERL for each floor of the building. The ELIN is used as the phone number for 911 callers within this ERL.

The figure below illustrates the use of ERLs and ELINs, with an E9-1-1 call from floor 2 at the branch office:



The table below shows an example of designating ERLs to physical areas (floors) in a building and associating each ERL with a unique ELIN.

Table 16-28:Designating ERLs and Assigning to ELINs

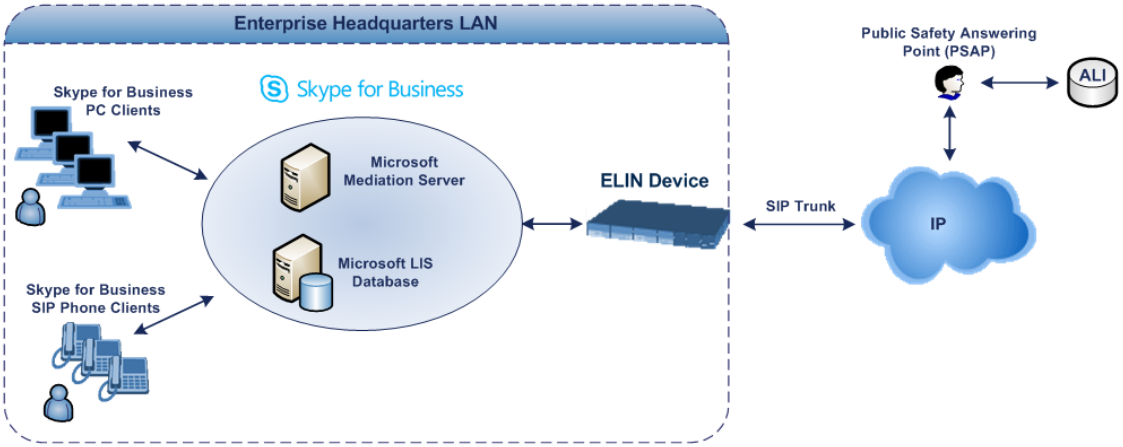
ERL Number	Physical Area	IP Address	ELIN
1	Floor 1	10.13.124.xxx	503 972-4410
2	Floor 2	10.15.xxx.xxx	503 972-4411
3	Floor 3	10.18.xxx.xxx	503 972-4412

In the table above, a unique IP subnet is associated per ERL. This is useful if you implement different subnets between floors. Therefore, IP phones, for example, on a specific floor are in the same subnet and therefore, use the same ELIN when dialing 9-1-1.

AudioCodes ELIN Device for Teams / Skype for Business E9-1-1 Calls to PSTN

Microsoft Mediation Server sends the location information of the E9-1-1 caller in the XML-based PIDF-LO body contained in the SIP INVITE message. However, this content cannot be sent on the SIP Trunksince they do not support such content. To solve this issue, Skype for Business requires a (ELIN SBC) to send the E9-1-1 call to the SIP Trunk. When Skype for Business sends the PIDF-LO to the , it parses the content and translates the calling number to an appropriate ELIN. This ensures that the call is routed to an appropriate PSAP, based on ELIN-address match lookup in the emergency service provider's ALI database.

The figure below illustrates an AudioCodes ELIN device deployed in the Skype for Business environment for handling E9-1-1 calls between the company and the emergency service provider.



Detecting and Handling E9-1-1 Calls

The ELIN device identifies E9-1-1 calls and translates their incoming E9-1-1 calling numbers into ELIN numbers, which are sent to the PSAP. The device handles the received E9-1-1 calls as follows:

1. The device identifies E9-1-1 calls if the incoming SIP INVITE message contains a PIDF-LO XML message body. This is indicated in the SIP *Content-Type* header, as shown below:

```
Content-Type: application/pidf+xml
```

2. The device extracts the ELIN number(s) from the ELIN field in the XML message. The ELIN field corresponds to the <CompanyName> column in the Location Information Server (LIS). The device supports up to five ELIN numbers per XML message. The ELINs are separated by a semicolon. The digits of the ELIN number can be separated by hyphens (xxx-xxx-xxx) or they can be adjacent (xxxxxxxxx), as shown below:

```
<ca:ELIN>1111-222-333; 1234567890 </ca:ELIN>
```

> For backward compatibility, if the ELIN field doesn't appear in the PIDF-LO, the device extracts the ELINs from the NAM field.

3. The device saves the From header value of the SIP INVITE message in its ELIN database table ('Call From' field). The ELIN table is used for PSAP callback, as discussed later in PSAP Callback for Dropped E9-1-1 Calls on page 379. The ELIN table also stores the following information:

   - **ELIN:** ELIN number
   - **Time:** Time at which the original E9-1-1 call was terminated with the PSAP

- **Count:** Number of E9-1-1 calls currently using the ELIN

An example of the ELIN database table is shown below:

ELIN	Time	Count	Index	Call From
4257275678	22:11:52	0	2	4258359333
4257275999	22:11:57	0	3	4258359444
4257275615	22:12:03	0	0	4258359555
4257275616	22:11:45	0	1	4258359777

The ELIN table stores this information for a user-defined period (see Configuring the E9-1-1 Callback Timeout), starting from when the E9-1-1 call, established with the PSAP, terminates. After this time expires, the table entry with its ELIN is disregarded and no longer used (for PSAP callback). Therefore, table entries of only the most recently terminated E9-1-1 callers are considered in the ELIN table. The maximum entries in the ELIN table is 300.

4. The device uses the ELIN number as the E9-1-1 calling number and sends it in the SIP INVITE (as an ANI / Calling Party Number) to the SIP Trunk.

An example of a SIP INVITE message received from an E9-1-1 caller is shown below. The SIP Content-Type header indicating the PIDF-LO and the ELIN field listing the ELINs are shown in **bold** typeface.

```
INVITE sip:911;phone-context=Redmond@192.168.1.12;user=phone SIP/2.0
From: "voip_911_user1"<sip:voip_911_
user1@contoso.com>;epid=1D19090AED;tag=d04d65d924
To: <sip:911;phone-context=Redmond@192.168.1.12;user=phone>
CSeq: 8 INVITE
Call-ID: e6828be1-1cdd-4fb0-bdda-cda7faf46df4
VIA: SIP/2.0/TLS 192.168.0.244:57918;branch=z9hG4bK528b7ad7
CONTACT: <sip:voip_911_
user1@contoso.com;opaque=user:epid:R4bCDaUj51a06PUbkraS0QAA;gruu>;te
xt;audio;video;image
PRIORITY: emergency
CONTENT-TYPE: multipart/mixed; boundary= ------=_NextPart_000_4A6D_
01CAB3D6.7519F890
geolocation: <cid:voip_911_user1@contoso.com>;inserted-by="sip:voip_911_
user1@contoso .com"
```

```
Message-Body:
------=_NextPart_000_4A6D_01CAB3D6.7519F890
Content-Type: application/sdp ; charset=utf-8
v=0
```

```
o=- 0 0 IN IP4 Client
s=session
c=IN IP4 Client
t=0 0
m=audio 30684 RTP/AVP 114 111 112 115 116 4 3 8 0 106 97
c=IN IP4 172.29.105.23
a=rtcp:60423
a=label:Audio
a=rtpmap:3 GSM/8000/1
a=rtpmap:101 telephone-event/8000
a=fmtp:101 0-16
a=rtpmap:0 PCMU/8000
a=rtpmap:8 PCMA/8000
a=ptime:20------=_NextPart_000_4A6D_01CAB3D6.7519F890
```

Content-Type: application/pidf+xml
Content-ID: <voip_911_user1@contoso.com>

```
<?xml version="1.0" encoding="utf-8"?>
```

```
<presence xmlns="urn:ietf:params:xml:ns:pidf"
xmlns:gp="urn:ietf:params:xml:ns:pidf:geopriv10"
xmlns:bp="urn:ietf:params:xml:ns:pidf:geopriv10:basicPolicy"
xmlns:ca="urn:ietf:params:xml:ns:pidf:geopriv10:civicAddr"
xmlns:ms="urn:schema:Rtc.LIS.msftE911PidfExtn.2008" entity="sip:voip_911_
user1@contoso.com"><tuple id="0"><status><gp:geopriv><gp:location-
info><ca:civicAddress><ca:country>US</ca:country><ca:A1>WA</ca:A1><ca:A3
>Redmond</ca:A3><ca:RD>163rd</ca:RD><ca:STS>Ave</ca:STS><ca:POD>N
E</ca:POD><ca:HNO>3910</ca:HNO><ca:LOC>40/4451</ca:LOC>
<ca:ELIN>1111-222-333; 1234567890 </ca:ELIN>
<ca:PC>98052</ca:PC></ca:civicAddress></gp:location-info><gp:usage-
rules><bp:retransmission-allowed>true</bp:retransmission-allowed></gp:usage-
rules></gp:geopriv><ms:msftE911PidfExtn><ms:ConferenceUri>sip:+142555501
99@contoso.com;user=phone</ms:ConferenceUri><ms:ConferenceMode>twowa
y</ms:ConferenceMode><LocationPolicyTagID
xmlns="urn:schema:Rtc.Lis.LocationPolicyTagID.2008">user-
tagid</LocationPolicyTagID
></ms:msftE911PidfExtn></status><timestamp>1991-09-
22T13:37:31.03</timestamp></tuple></presence>
```

```
------=_NextPart_000_4A6D_01CAB3D6.7519F890--
```

Pre-empting Existing Calls for E9-1-1 Calls

If the ELIN device receives an E9-1-1 call from the IP network and there are unavailable channels (for example, all busy), the device immediately terminates one of the non-E9-1-1 calls (arbitrary) and accepts the E9-1-1 call on the freed-up channel:

- Preemption is done only on a call belonging to the same source IP Group from which the E9-1-1 call is received, or the same destination IP Group (i.e., PSAP Server).

This feature is initiated only if the received SIP INVITE message contains a Priority header set to "emergency", as shown below:

```
Priority: emergency
```

PSAP Callback for Dropped E9-1-1 Calls

As the E9-1-1 service automatically provides all the contact information of the E9-1-1 caller to the PSAP, the PSAP operator can call back the E9-1-1 caller. This is especially useful in cases where the caller disconnects prematurely. However, as the Enterprise sends ELINs to the PSAP for E9-1-1 calls, a callback can only reach the original E9-1-1 caller using the device to translate the ELIN number back into the E9-1-1 caller's extension number.

In the ELIN table of the device, the temporarily stored *From* header value of the SIP INVITE message originally received from the E9-1-1 caller is used for PSAP callback. When the PSAP makes a callback to the E9-1-1 caller, the device translates the called number (i.e., ELIN) received from the PSAP to the corresponding E9-1-1 caller's extension number as matched in the ELIN table.

The handling of PSAP callbacks by the device is as follows:

1. When the device receives a call from the emergency service provider, it searches the ELIN table for an ELIN that corresponds to the received called party number in the incoming message.
2. If a match is found in the ELIN table, it routes the call to the Mediation Sever by sending a SIP INVITE, where the values of the *To* and *Request-URI* are taken from the value of the original *From* header that is stored in the ELIN table (in the **Call From** column).
3. The device updates the 'Time' field in the ELIN table (the 'Count' field is not affected).

The PSAP callback can be done only within a user-defined period (see Configuring the E9-1-1 Callback Timeout), started from after the original E9-1-1 call, established with the PSAP is terminated. After this time expires, the table entry with its ELIN is disregarded and no longer used (for PSAP callback). Therefore, table entries of only the most recently terminated E9-1-1 callers are considered in the ELIN table. If the PSAP callback is done after this timeout expires, the device is unable to route the call to the E9-1-1 caller and instead, either sends it as a regular call or most likely, rejects it if there are no matching routing rules. However, if another E9-1-1 caller has subsequently been processed with the same ELIN number, the PSAP callback is routed to this new E9-1-1 caller.

In scenarios where the same ELIN number is used by multiple E9-1-1 callers, upon receipt of a PSAP callback, the device sends the call to the most recent E9-1-1 caller. For example, if the ELIN number "4257275678" is being used by three E9-1-1 callers, as shown in the table below, then when a PSAP callback is received, the device sends it to the E9-1-1 caller with phone number "4258359555".

Table 16-29:Choosing Caller of ELIN

ELIN	Time	Call From
4257275678	11:00	4258359333
4257275678	11:01	4258359444
4257275678	11:03	**4258359555**

Selecting ELIN for Multiple Calls within Same ERL

The device supports the receipt of up to five ELIN numbers in the XML message of each incoming SIP INVITE message. As discussed in the preceding sections, the device sends the ELIN number as the E9-1-1 calling number to the emergency service provider. If the XML message contains more than one ELIN number, the device chooses the ELIN according to the following logic:

- If the first ELIN in the list is not being used by other active calls, it chooses this ELIN.
- If the first ELIN in the list is being used by another active call, the device skips to the next ELIN in the list, and so on until it finds an ELIN that is not being used and sends this ELIN.
- If all the ELINs in the list are in use by active calls, the device selects the ELIN number as follows:
  a. The ELIN with the lowest count (i.e., lowest number of active calls currently using this ELIN).
  b. If the count between ELINs is identical, the device selects the ELIN with the greatest amount of time passed since the original E9-1-1 call using this ELIN was terminated with the PSAP. For example, if E9-1-1 caller using ELIN 4257**2**75678 was terminated at **11:01** and E9-1-1 caller using ELIN 425727567**0** was terminated at **11:03**, then the device selects ELIN 425727567**8**.

  In this scenario, multiple E9-1-1 calls are sent with the same ELIN.

Configuring AudioCodes ELIN Device

This section describes E9-1-1 configuration of the AudioCodes ELIN Gateway deployed in the Microsoft Teams / Skype for Business environment.

Enabling the E9-1-1 Feature

By default, the ELIN device feature for E9-1-1 emergency call handling in a Microsoft Teams / Skype for Business environment is disabled.

➢ To enable ELIN feature for the SBC application:

- For the IP Group through which you want to communicate with the public-safety answering point (PSAP), configure the 'SBC PSAP Mode' parameter to **Enable**. For more information, see Configuring IP Groups.

Configuring the E9-1-1 Callback Timeout

If the initial established call between the E9-1-1 caller and the PSAP is prematurely terminated, the PSAP can use the ELIN to call back the E9-1-1 caller within a user-defined time interval (in minutes) from when the call was terminated. By default, an ELIN can be used for PSAP callback within 30 minutes after the call terminates. You can change this to any value between 0 and 60. For more information on PSAP callback for dropped E9-1-1 calls, see PSAP Callback for Dropped E9-1-1 Calls on page 379.

➢ To configure the E9-1-1 callback timeout

1. Open the Priority & Emergency page (**Setup** menu > **Signaling & Media** tab > **SIP Definitions** folder > **Priority and Emergency**).
2. In the 'E911 Callback Timeout' field (E911CallbackTimeout), enter the required callback timeout.

3. Click **Apply**.

Configuring SBC IP-to-IP Routing Rule for E9-1-1

To route incoming Teams / Skype for Business E9-1-1 calls to the emergency service provider's PSAP server, you need to configure routing rules in the IP-to-IP Routing table for routing between the emergency callers' IP Group and the PSAP server's IP Group. The only special configuration is to define the emergency number (e.g., 911) in the 'Destination Username Pattern' parameter of the IP Group belonging to the E9-1-1 callers. The following example shows IP-to-IP routing rules for E9-1-1:

INDEX	NAME	ROUTING POLICY	ALTERNATIVE ROUTE OPTIONS	SOURCE IP GROUP	REQUEST TYPE	SOURCE USERNAME PREFIX	DESTINATION USERNAME PREFIX	DESTINATION TYPE	DESTINATION IP GROUP
0	E911 > PSAP	Default_SBCRou	Route Row	LAN IP PBX	All	*	911	IP Group	PSAP Server
1	PSAP > E911	Default_SBCRou	Route Row	PSAP Server	All	*	*	IP Group	LAN IP PBX

Viewing the ELIN Table

To view the ELIN table:

- CLI:

```
# show voip e911
ELIN            Time     Count Index Call From
-------------------------------------------------------------
4257275678       22:11:52 0    2    4258359333
4257275999       22:11:57 0    3    4258359444
257275615       22:12:03 0    0    4258359555
4257275616       22:11:45 0    1    4258359777
------------ Current Time: 22:12:40
```

- Syslog, by invoking the following Web command shell:

```
SIP / GateWay / E911Dump
```

Microsoft Skype for Business Presence of Third-Party Endpoints

Microsoft presence capability allows Skype for Business users to know the status (e.g., "Available" or "Do Not Disturb") of their contacts. Presence status of contacts is displayed on the user's Skype for Business endpoint. Presence information of Skype for Business endpoints (such as Skype for Business desktop client) is handled solely by the Skype for Business Server, without any intervention of the device. However, when third-party (non-Skype for Business) endpoint devices (e.g., mobile phone or PBX phone) are used by the Skype for Business users, presence status information can only be reported to the Skype for Business Server by the device. For example, if John and Alice are Skype for Business users and John makes or receives a call on a mobile device, Alice is able to see that John is in a call, even though the call is not on a native Skype for Business endpoint. Once the device reports the presence status, the Skype for Business Server sends this status change to the Skype for Business users in the network.

- Currently, the device reports the following presence status:
  - ✓ "On the Phone" - user is busy (in a call or doesn't want to be disturbed)
  - ✓ "Clear" - cancels the "On the Phone" status (returning the user's presence to its previous state)
- The feature supports Skype for Business Server 2015 and Lync Server version 5.0.8308.866 and later.

The device notifies the Skype for Business Server of a user's presence status, by using SIP PUBLISH messages. The message transactions between the device and Skype for Business Server is as follows:

1. The device routes a call between two Skype for Business users and when connected, sends a PUBLISH message with the Event header set to "presence", Expires header set to "600",

Content-Type header set to "application/pidf+xml", and where the XML body's "activity" is set to "on-the-phone", as shown in the following example for user John Doe:

```
PUBLISH sip:john.doe@sfb.example SIP/2.0
From: <sip:john.doe@sfb.example>;tag=1c537837102
To: <sip:john.doe@sfb.example>
CSeq: 1 PUBLISH
Event: presence
Expires: 600
Content-Type: application/pidf+xml
Content-Length: 489
```

```
<?xml version="1.0" encoding="utf-8"?>
<presence xmlns="urn:ietf:params:xml:ns:pidf"
xmlns:ep="urn:ietf:params:xml:ns:pidf:status:rpid-status"
xmlns:et="urn:ietf:params:xml:ns:pidf:rpid-tuple"
xmlns:ci="urn:ietf:params:xml:ns:pidf:cipid"
entity="sip:john.doe@sfb.example">
<tuple id="0">
<status>
<basic>open</basic>
<ep:activities>
<ep:activity>on-the-phone</ep:activity>
</ep:activities>
</status>
</tuple>
<ci:display-name>John Doe</ci:display-name>
</presence>
```

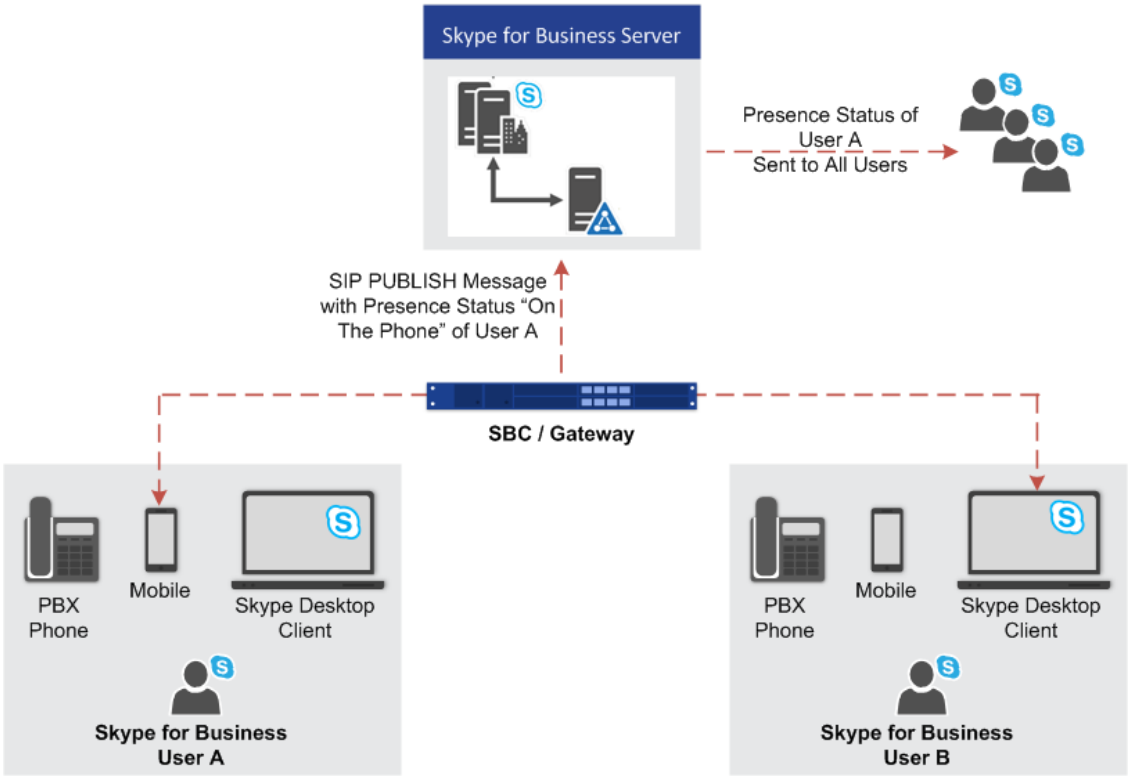
2. The Skype for Business Server responds to the device with a SIP 200 OK. The message is sent with a SIP-ETag header which identifies the entity (and Expires header set to 600 seconds), as shown in the following example:

```
SIP/2.0 200 OK
From: "John Doe"<sip:john.doe@sfb.example>;tag=1c537837102
To:
<sip:john.doe@sfb.example>;tag=0E4324A4B27040E4A167108D4FAD27E3
Call-ID: 12848966432792016357360@10.33.221.57
CSeq: 1 PUBLISH
Via: SIP/2.0/TLS 10.33.221.57:5061;alias;…received=10.33.221.57;ms-
received port=48093;ms-received-cid=4900
SIP-ETag: 2545777538-1-1
Expires: 600
Content-Length: 0
```

3. If the call lasts longer than 600 seconds, the device sends another PUBLISH message with the same SIP-ETag value and with an Expires header value of 600 seconds. The Skype for Business Server responds with another 200 OK, but with a new SIP-ETag value (and Expires header set to 600 seconds). This scenario occurs for each 600-second call interval.
4. When the call ends, the device sends a PUBLISH message to cancel the user's online presence status (and the user's previous presence state is restored). The message is sent with a SIP-If-Match header set to the matching entity tag (SIP-ETag) value (i.e., SIP-ETag value of last 200 OK) and Expires header value set to "0", as shown in the following example:

```
PUBLISH sip:john.doe@sfb.example SIP/2.0
From: <sip:john.doe@sfb.example>;tag=1c1654434948
To: <sip:john.doe@sfb.example>
CSeq: 1 PUBLISH
Contact: <sip:john.doe@10.33.221.57:5061;transport=tls>
Event: presence
Expires: 0
User-Agent: sur1-vg1.ecarecenters.net/v.7.20A.001.080
SIP-If-Match: 2545777538-1-1
Content-Length: 0
```

The following figure shows a basic illustration of the device's integration into Microsoft Skype for Business Presence feature for third-party endpoints.



Configuring Skype for Business Server for Presence

On the Skype for Business Server side, you need to define the device in the Skype for Business Topology as a Trusted Application.

- Detailed configuration of Skype for Business Server is beyond the scope of this document.
- Before performing the below procedure, make sure that you have defined the device in the PSTN Gateway node of the Skype for Business Server Topology (using the Topology Builder).

Using the Skype for Business Server Management Shell, perform the following steps:

1. **Obtain the Site ID**

   Run the following cmdlet to retrieve the SiteId property of the site:

   ```
   Get-CsSite
   ```

2. **Create a Trusted Application Pool**

   Run the following cmdlet to create a new pool to host the presence application:

   ```
   New-CsTrustedApplicationPool -Identity <Pool FQDN> -Registrar <Registrar FQDN> -Site <Site Id>
   ```

   where:

   - *Identity* is the FQDN of the device, which sends the SIP PUBLISH messages with the presence status to Skype for Business Server
   - *Registrar* is the FQDN of the Registrar service for the pool
   - *Site* is the Site Id

   For example:

   ```
   New-CsTrustedApplicationPool -Identity audcsbcgw.example.com -Registrar skypepool.example.com -Site Portland
   ```

3. **Add the Trusted Application (Presence) to the Pool**

   ```
   New-CsTrustedApplication-ApplicationId <String> -TrustedApplicationPoolFqdn <String> -Port <Port Number>
   ```

   where:

   - *ApplicationId* is the name of the application
   - *TrustedApplicationPoolFqdn* is the FQDN of the trusted application pool

- *Port* is the port number on which the application will run (5061)

For example:

```
New-CsTrustedApplication -ApplicationId MSpresence -
TrustedApplicationPoolFqdn audcsbcgw.example.com  -Port 5061
```

Make sure the port number matches the port number configured on the device.

4. **Enable and Publish the Skype for Business Server 2015 Topology**

   Run the following cmdlet to publish and enable your new topology:

```
Enable-CsTopology
```

Configuring the Device for Skype for Business Presence

The following procedure describes how to configure the device for notifying Skype for Business Server of presence status of Skype for Business users when making and receiving calls using third-party, endpoint devices. To help you understand the configuration, the following lists in chronological order the main processing steps:

1. The device receives an incoming call.
2. The device uses a Call Setup Rule to perform LDAP queries on the Microsoft Active Directory to retrieve Skype for Business usernames (Request URIs) for the corresponding calling (source) and/or called (destination) number. For SBC calls, the Call Setup Rule is associated with the classified source IP Group (in the IP Groups table).
3. The device routes the call to the required destination, according to the normal routing rules.
4. When the call is connected, the device sends a SIP PUBLISH message to Skype for Business Server, indicating that the users' presence status is now "On-the-Phone".
5. When the call ends, the device sends another SIP PUBLISH message to the Skype for Business Server, clearing the users' "On-the-Phone" status (the presence status changes to what it was before the call was connected).

➢ **To configure the device for Skype for Business presence:**

1. Enable the Microsoft presence feature: open the SIP Definitions General Settings page (**Setup** menu > **Signaling & Media** tab > **SIP Definitions** folder > **SIP Definitions General Settings**), and then from the 'Microsoft Presence Status' drop-down list, select **Enable**:

2. Configure a TLS Context (TLS certificate) for secured communication (mutual authentication) between the device and the Skype for Business Server (see Configuring TLS Certificate Contexts).
3. Configure a Proxy Set to define the address of the Skype for Business Server (see Configuring Proxy Sets). Make sure you configure the following:
   - 'TLS Context Name': Assign the TLS Context that you configured in Step 2 (above).
   - 'Proxy Address': Configure the address (FQDN or IP address).
   - 'Transport Type': **TLS**
4. Configure an IP Group to represent the Skype for Business Server (see Configuring IP Groups). Make sure that you assign it with the Proxy Set that you configured in Step 3 (above).
5. Assign the IP Group of the Skype for Business Server as the destination (presence gateway) to where the device must send the PUBLISH messages: open the SIP Definitions General Settings page (**Setup** menu > **Signaling & Media** tab > **SIP Definitions** folder > **SIP Definitions General Settings**), and then in the 'Presence Publish IP Group ID' field, enter the IP Group ID of the Skype for Business Server that you configured in Step 4 (above):

6. Configure the Skype for Business LDAP server (Active Directory) to query for the Skype for Business users' SIP URIs (see Configuring LDAP Servers).
7. Configure Call Setup Rules to perform LDAP queries in the Microsoft Active Directory for the SIP URI of the caller (source) and called (destination) parties (see Configuring Call Setup Rules). The device first needs to search the AD for the caller or called number of the third-party endpoint device. For example, to search for a called mobile number, the searched LDAP Attribute would be "mobile" set to the value of the destination number (e.g., 'mobile=+' + param.call.dst.user). If the entry exists, the query searches for the Attribute (e.g., ipPhone) where the SIP URI is defined for the corresponding mobile user. If found, the query returns the Attribute's value (i.e., URI) to the device (instructed using the special 'Condition' string "presence.dst" or "presence.src"). This is the URI that the device uses as the Request-URI in the PUBLISH message that it sends to the Skype for Business Server. The configuration of the example used in this step is shown below:

Parameter	Rule 1	Rule 2
'Request Type'	**LDAP**	**LDAP**
'Request Key'	'mobile=+' + param.call.dst.user	'mobile=+' + param.call.src.user
'Attributes To Get'	ipPhone	ipPhone
'Condition'	ldap.attr.ipPhone exists	ldap.attr.ipPhone exists

Parameter	Rule 1	Rule 2
'Action Subject'	presence.dst	presence.src
'Action Type'	**Add**	**Add**
'Action Value'	ldap.attr.ipPhone	ldap.attr.ipPhone

8. Configure routing rules to route the calls in the network.
9. Configure IP Groups to represent your call party entities, and assign them the group of Call Setup Rules (Set ID) that you configured in Step 7 (above). For configuring IP Groups, see Configuring IP Groups.

Microsoft Teams with Local Media Optimization

The device can be configured to support the Local Media Optimization feature when deployed in a Microsoft Teams environment. This feature is intended for complex environments consisting of a central SBC device (i.e., this device that you are configuring), which is referred to by Microsoft as the *Proxy SBC*, integrated in the Teams environment, and multiple remote SBCs or Gateways (referred to by Microsoft as *remote site SBCs*). In this environment, the central SBC determines the optimal path for connecting calls between the Teams clients, based on network connectivity (good or bad) and voice quality. The device path selection is based on supplementary information provided by Microsoft using their proprietary headers that are included in the SIP messages during call setup between Teams clients:

Microsoft SIP Header	Value	Description
X-MS-UserLocation	Internal or External	Indicates if the Teams client is located in the internal or external network with respect to the central SBC. Based on the header value, the device selects the Media Realm, using the IP Group's 'Internal Media Realm' or 'Media Realm' parameters, respectively.
X-MS-MediaPath	sbc1.contoso.com sbc2.contoso.com ...	Indicates the order of remote SBCs that should be used for the media path between the Teams clients. If the first address is the central SBC itself, the media traverses the device (non-direct media).

Configuration of the device for Local Media Optimization is done on the IP Group of the Teams client, using the following IP Group table parameters:

- 'Teams Local Media Optimization Handling': This parameter enables Local Media Optimization and defines how the device handles the Teams call based on the Microsoft proprietary SIP headers.
- 'Internal Media Realm': Assigns a Media Realm which is used if the X-MS-UserLocation header value is "Internal". If the header value is "External" (or not present), the Media Realm assigned by the 'Media Realm' parameter is used.
- 'Teams Local Media Optimization Initial Behavior': This parameter defines how the central SBC device initially sends the received INVITE message with the SDP Offer to Teams.

For more information on the above parameters, see their descriptions in Configuring IP Groups on page 451.

For detailed technical information on deploying the device in a Microsoft Teams environment with Local Media Optimization, contact your AudioCodes sales representative.

17 Quality of Experience

This chapter describes how to configure the Quality of Experience feature.

Reporting Voice Quality of Experience to OVOC

The device can be configured to report voice (media) Quality of Experience (QoE) to AudioCodes' One Voice Operations Center (OVOC). The reports include real-time metrics of the quality of the actual call experience, which are then processed by OVOC.

OVOC is also a VoIP-quality monitoring and analysis tool. It provides comprehensive details on voice traffic quality, allowing system administrators to quickly identify, fix and prevent issues that could affect the voice calling experience in enterprise and service provider VoIP networks. IT managers and administrators can employ OVOC in their VoIP networks to guarantee effective utilization, smooth performance, reliable QoS levels, and SLA fulfillment.

- For information on OVOC, refer to the *OVOC User's Manual*.
- For configuring the SNMP connection between the device and OVOC, see Configuring SNMP for OVOC Connectivity on page 89.

Configuring OVOC for Quality of Experience

The Quality of Experience Settings table lets you configure the address (and other connectivity parameters) of AudioCodes One Voice Operations Center (OVOC) server to where the device sends Quality of Experience (QoE) voice metric reports.

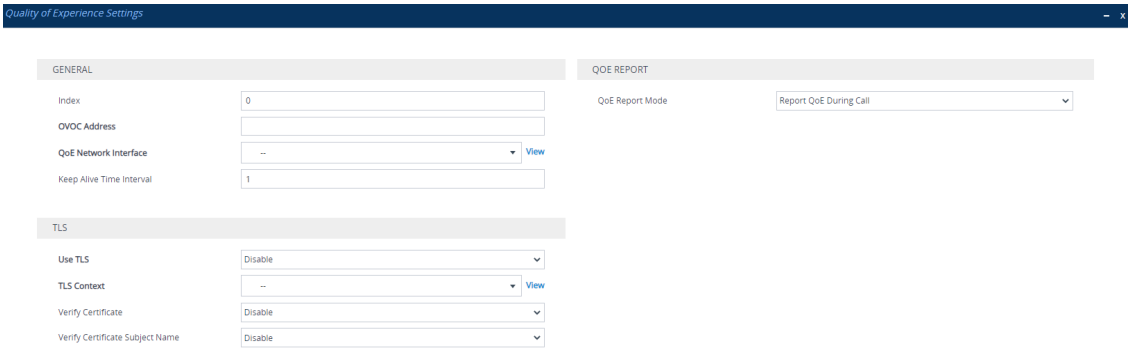
You can also configure the device to use a TLS connection with OVOC. Before you can do this, configure a TLS Context (certificate) in the TLS Contexts table (see Configuring TLS Certificate Contexts). If no TLS Context is specified, the device uses the default TLS Context (ID 0). You can also configure at what stage of the call the device sends the QoE report to OVOC. The report can be sent during the call or only at the end of the call. Reporting at the end of the call may be beneficial when there is network congestion as this reduces bandwidth usage over time.

If a QoE traffic overflow is experienced between OVOC and the device, the device sends the QoE data only at the end of the call, regardless of your settings.

The following procedure describes how to configure the OVOC server for QoE through the Web interface. You can also configure it through ini file [QOESettings] or CLI (`configure voip > qoe qoe-settings`).

➢ **To configure the OVOC server for QoE:**

1. Open the Quality of Experience Settings table (**Setup** menu > **Signaling & Media** tab > **Media** folder > **Quality of Experience** > **Quality of Experience Settings**).



2. Configure the OVOC server according to the parameters described in the table below.
3. Click **Apply**, and then reset the device with a save-to-flash for your settings to take effect.

Table 17-1: Quality of Experience Settings Parameter Descriptions

Parameter	Description
General	
'Index' `tls` [QOESettings_Index]	Defines an index number for the new table row. **Note:** Each row must be configured with a unique index.
'OVOC Address' `server-name` [QOESettings_ServerName]	Defines the address of the OVOC server to where the QoE reports are sent. The valid value is an IP address (IPv4) or an FQDN (hostname). **Note:** If you are using a WebSocket tunnel connection between the device and OVOC, then configure the parameter to the IP address mentioned in Configuring WebSocket Tunnel with OVOC on page 91.
'QoE Network Interface' `interface` [QOESettings_Interface]	Assigns an IP network interface from which the device sends the QoE reports. The default is the OAMP interface (**O+M+C**). To configure network interfaces, see Configuring IP Network Interfaces.
'Keep Alive Time Interval' `keep-alive-time` [QOESettings_KeepAliveTime]	Defines the interval (in seconds) between every consecutive keep-alive message that the device sends to the OVOC server. Keep-alive messages can be useful to keep the communication link between the device and OVOC open when there is no other traffic flow between them. The default is 1. A value of 0 disables the keep-alive feature.

Parameter	Description
TLS	
'Use TLS' `tls` [QOESettings_EnableTls]	Enables a TLS connection with the OVOC server. ■ [0] **Disable** (default) ■ [1] **Enable**
'TLS Context' `tls-context-name` [QOESettings_ContextName]	Assigns a TLS Context (certificate) for the TLS connection with the OVOC server. The default is the default TLS Context (ID 0). **Note:** The parameter is applicable only if the 'Use TLS' parameter is configured to **Enable**.
'Verify Certificate' `verify-certificate` [QOESettings_VerifyCertificate]	Enables the verification of the TLS certificate that is used in the incoming connection request from the OVOC server. ■ [0] **Disable** = (Default) No certificate verification is done. ■ [1] **Enable** = The device verifies the authentication of the certificate received from the OVOC server. The device authenticates the certificate against the trusted root certificate store associated with the assigned TLS Context and if ok, allows communication with OVOC. If authentication fails, the device denies communication (i.e., handshake fails). The device can also authenticate the certificate by querying with an Online Certificate Status Protocol (OCSP) server whether the certificate has been revoked. This is configured for the assigned TLS Context. **Note:** The parameter is applicable only if the 'Use TLS' parameter is configured to **Enable**.
'Verify Certificate Subject Name' `verify-certificate-subject-name` [QOESettings_ VerifyCertificateSubjectName]	Enables the verification of the TLS certificate subject name (Common Name / CN or Subject Alternative Name / SAN) that is used in the incoming connection request from the OVOC server. ■ [0] **Disable** = (Default) No verification is done. ■ [1] **Enable** = The device verifies the subject name of the certificate received from the OVOC server with the hostname or IP address configured for

Parameter	Description
	OVOC (in the 'OVOC Address' parameter above). If authentication fails, the device denies communication (i.e., handshake fails). **Note:** The parameter is applicable only if the 'Use TLS' parameter is configured to **Enable**.
QoE Report	
'QoE Report Mode' `report-mode` [QOESettings_ReportMode]	Defines at what stage of the call the device sends the call's QoE data to the OVOC server. ■ [0] **Report QoE During Call** (default) ■ [1] **Report QoE at End of Call** **Note:** If a QoE traffic overflow between OVOC and the device occurs, the device sends the QoE data only at the end of the call, regardless of the parameter's settings.

Configuring Clock Synchronization between Device and OVOC

To ensure accurate call quality statistics and analysis by OVOC, you must configure the device and the OVOC server with the same clock source for clock synchronization. In other words, you need to configure them with the same NTP server (same address).

The NTP server can be one of the following:

- OVOC server (also acting as an NTP server)
- Third-party, external NTP server

To configure, the NTP server's address on the device, see Configuring Automatic Date and Time using SNTP.

Configuring Firewall Rules for OVOC Traffic

To allow incoming traffic from OVOC, you need to configure the device's firewall (Firewall table) with additional "Allow" firewall rules, as described in Configuring Firewall Rules to Allow Incoming OVOC Traffic on page 184.

Enabling RTCP XR Reporting to OVOC

For the device to be able to send voice metric reports to AudioCodes OVOC, you need to enable the RTP Control Protocol Extended Reports (RTCP XR) VoIP management protocol. RTCP XR defines a set of voice metrics that contain information for assessing VoIP call quality and

diagnosing problems. Enabling RTCP XR means that the device can send RTCP XR messages, containing the call-quality metrics, to OVOC.

For enabling RTCP XR reporting, see Configuring RTCP XR. To configure what to report to OVOC, see Configuring Quality of Experience Profiles.

Configuring Quality of Experience Profiles

Quality of Experience Profiles enable you to effectively monitor the quality of voice calls traversing the device in your network. Quality of Experience Profiles define severity thresholds for voice metrics monitored by the device, which if crossed can result in various actions (discussed later in the section).

Quality of Experience is configured using two tables with parent-child type relationship. The Quality of Experience Profile table is the parent, which defines the name of the Quality of Experience Profile. The Quality of Experience Color Rules table is the child, which defines severity thresholds per voice metric for the specific Quality of Experience Profile. You can configure up to 256 Quality of Experience Profiles and up to 256 Quality of Experience Color Rules.

Once configured, you can apply the Quality of Experience Profiles to specific calls (network links), by assigning them to any of the following configuration entities:

- IP Groups (see Configuring IP Groups)
- Media Realms (see Configuring Media Realms)
- Remote Media Subnets (see Configuring Remote Media Subnets)

The Quality of Experience Profile allows you to configure thresholds for the following monitored voice metrics:

- **Mean Opinion Score (MOS):** MOS is the average grade on a quality scale, expressed as a single number in the range of 1 to 5, where 1 is the lowest audio quality and 5 the highest audio quality.
- **Delay (or latency):** Time it takes for information to travel from source to destination (round-trip time).
- **Packet Loss:** Lost packets are RTP packets that are not received by the voice endpoint. Packet loss can result in choppy voice transmission.
- **Jitter:** Jitter can result from uneven delays between received voice packets. To space evenly, the device's jitter buffer adds delay. The higher the measurement, the greater the impact of the jitter buffer's delay on audio quality.
- **Residual Echo Return Loss (RERL):** An echo is a reflection of sound arriving at the listener at some time after the sound was initiated (often by the listener). Echo is typically caused by delay.

At any given time during a call, a voice metric can be in one of the following color-coded quality states (as displayed in OVOC):

- **Green:** Indicates good call quality
- **Yellow:** Indicates fair call quality
- **Red:** Indicates poor call quality

When the threshold of a voice metric is crossed, the device changes the alarm severity and corresponding color-coded quality state of the call:

- **Minor Threshold (Yellow):** Lower threshold that indicates changes from Green or Red to Yellow.
- **Major Threshold (Red):** Higher threshold that indicates changes from Green or Yellow to Red.

The device also uses hysteresis to determine whether the threshold has indeed being crossed. Hysteresis defines the amount of fluctuation from the threshold in order for the threshold to be considered as crossed (i.e., change in color state). Hysteresis is used to avoid false reports being sent by the device. Hysteresis is used only for threshold crossings toward a lesser severity (i.e., from Red to Yellow, Red to Green, or Yellow to Green).

The following example is used to explain how the device considers threshold crossings. The example is based on the MOS of a call, where the Major threshold is configured to 2, the Minor threshold to 4 and the hysteresis for both thresholds to 0.1:

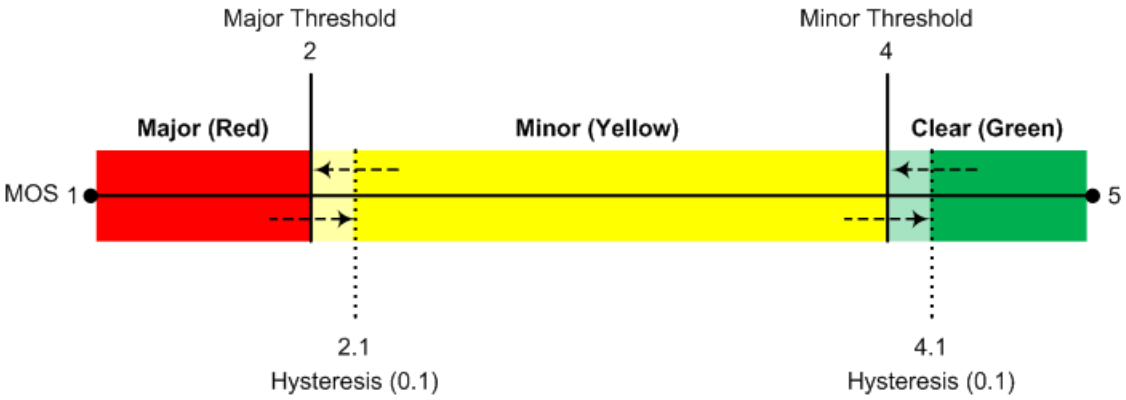


Table 17-2: Threshold Crossings based on Threshold and Hysteresis

Threshold Crossing	Calculation	Threshold based on Example
Green to Yellow (Minor alarm)	The change occurs if the measured metric crosses the configured Minor threshold only (i.e., hysteresis is not used).	4
Green to Red (Major alarm)	The change occurs if the measured metric crosses the configured Major threshold only (i.e., hysteresis is not used).	2

Threshold Crossing	Calculation	Threshold based on Example
Yellow to Red (Major alarm)	The change occurs if the measured metric crosses the configured Major threshold only (i.e., hysteresis is not used).	2
Red to Yellow (Minor alarm)	The change occurs if the measured metric crosses the configured Major threshold with hysteresis configured for the Major threshold.	2.1 (i.e., 2 + 0.1)
Red to Green (alarm cleared)	The change occurs if the measured metric crosses the configured Minor threshold with hysteresis configured for the Minor threshold.	4.1 (i.e., 4 + 0.1)
Yellow to Green (alarm cleared)	The change occurs if the measured metric crosses the configured Minor threshold with hysteresis configured for the Minor threshold.	4.1 (i.e., 4 + 0.1)

Each time a voice metric threshold is crossed (i.e., color changes), the device can do the following depending on configuration:

- Report the change in the measured metrics to AudioCodes' OVOC. OVOC displays this call quality status for the associated link (IP Group, Media Realm, or Remote Media Subnet). To configure the OVOC's address, see Configuring the SEM Server.
- Depending on the crossed threshold type, you can configure the device to reject calls to the destination IP Group or use an alternative IP Profile for the IP Group. For more information, see Configuring Quality of Service Rules.
- Alternative routing based on measured metrics. If a call is rejected because of a crossed threshold, the device generates a SIP 806 response. You can configure this SIP response code as a reason for alternative routing (see Configuring SIP Response Codes for Alternative Routing Reasons).

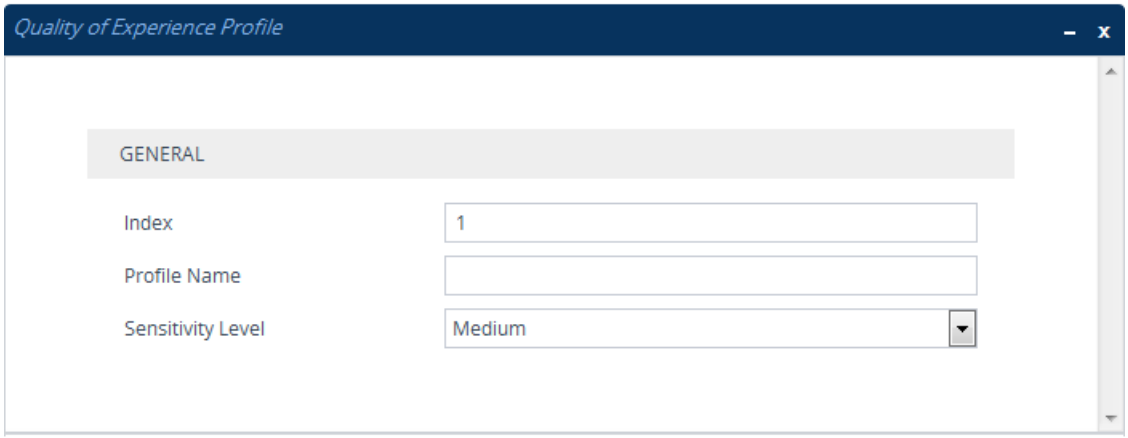
For your convenience, the device provides pre-configured Quality of Experience Profiles. One of these pre-configured profiles is the default Quality of Experience Profile, which is used if you do not configure a Quality of Experience Profile.

The following procedure describes how to configure Quality of Experience Profiles through the Web interface. You can also configure it through other management platforms:

- **Quality of Experience Profile table:** *ini* file [QoEProfile] or CLI (`configure voip > qoe qoe-profile`)
- **Quality of Experience Color Rules table:** *ini* file [QOEColorRules] or CLI (`configure voip > qoe qoe-color-rules`)

➢ **To configure a QoE Profile:**

1. Open the Quality of Experience Profile table (**Setup** menu > **Signaling & Media** tab > **Media** folder > **Quality of Experience** > **Quality of Experience Profile**).
2. Click **New**; the following dialog box appears:

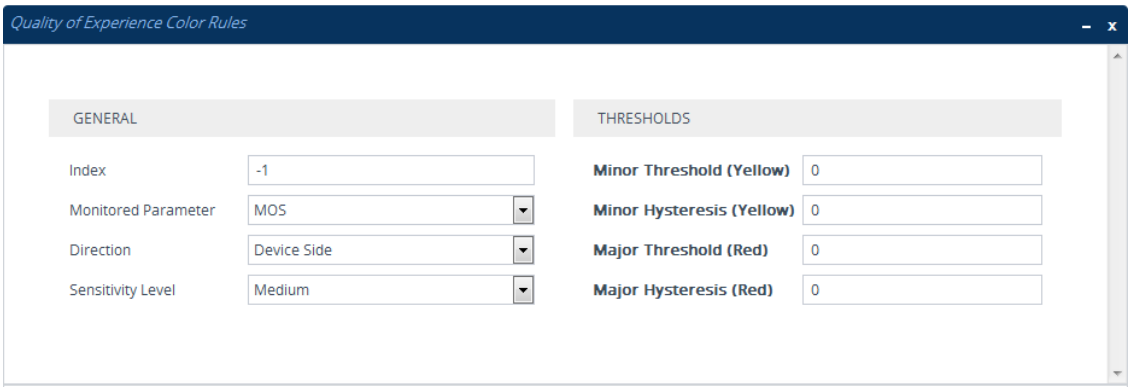


3. Configure a QoE Profile according to the parameters described in the table below.
4. Click **Apply**.

Table 17-3: Quality of Experience Profile Table Parameter Descriptions

Parameter	Description
'Index' [QOEProfile_Index]	Defines an index number for the new table row. **Note:** Each row must be configured with a unique index.
'Profile Name' `name` [QOEProfile_Name]	Defines a descriptive name, which is used when associating the row in other tables. The valid value is a string of up to 20 characters. **Note:** The parameter value cannot contain a forward slash (/).
'Sensitivity Level' `sensitivity-level` [QOEProfile_ SensitivityLevel]	Defines the pre-configured threshold profile to use. ■ [0] **User Defined** = Need to define thresholds per monitored parameter in the Quality of Experience Color Rules table. ■ [1] **Low** = Pre-configured low sensitivity thresholds. ■ [2] **Medium** = (Default) Pre-configured medium sensitivity thresholds. ■ [3] **High** = Pre-configured high sensitivity thresholds. Reporting is done for small fluctuations in parameter values.

5. In the Quality of Experience Profile table, select the row for which you want to configure QoE thresholds, and then click the **Quality of Experience Color Rules** link located below the table; the Quality of Experience Color Rules table appears.
6. Click **New**; the following dialog box appears:



7. Configure a rule according to the parameters described in the table below.
8. Click **New**, and then save your settings to flash memory.

Table 17-4: Quality of Experience Color Rules Table Parameter Descriptions

Parameter	Description
General	
'Index' `index` [QOEColorRules_ ColorRuleIndex]	Defines an index number for the new table row. **Note:** Each row must be configured with a unique index.
'Monitored Parameter' `monitored-parameter` [QOEColorRules_ monitoredParam]	Defines the parameter to monitor and report. ■ [0] **MOS** (default) ■ [1] **Delay** ■ [2] **Packet Loss** ■ [3] **Jitter** ■ [4] **RERL [Echo]**
'Direction' `direction` [QOEColorRules_ direction]	Defines the monitoring direction. ■ [0] **Device Side** (default) ■ [1] **Remote Side**
'Sensitivity Level' `sensitivity-`	Defines the sensitivity level of the thresholds. ■ [0] **User Defined** = Need to define the thresholds in the

Parameter	Description
`level` [QOEColorRules_profile]	parameters described below. ■ [1] **Low** = Pre-configured low sensitivity threshold values. Thus, reporting is done only if changes in parameters' values are significant. ■ [2] **Medium** = (Default) Pre-configured medium sensitivity threshold values. ■ [3] **High** = Pre-configured high sensitivity threshold values. Thus, reporting is done for small fluctuations in parameter values.
Thresholds	
'Minor Threshold (Yellow)' `minor-threshold-yellow` [QOEColorRules_MinorThreshold]	Defines the Minor threshold value, which is the lower threshold located between the Yellow and Green states. To consider a threshold crossing: ■ **Increase in severity (i.e., Green to Yellow):** Only this value is used. ■ **Decrease in severity (Red to Green, or Yellow to Green):** This value is used with the hysteresis, configured by the 'Minor Hysteresis (Yellow)' parameter (see below). The valid threshold values are as follows: ■ MOS values are in multiples of 10. For example, to denote a MOS of 3.2, the value 32 (i.e., 3.2*10) must be entered. ■ Delay values are in msec. ■ Packet Loss values are in percentage (%). ■ Jitter is in msec. ■ Echo measures the Residual Echo Return Loss (RERL) in dB.
'Minor Hysteresis (Yellow)' `minor-hysteresis-yellow` [QOEColorRules_MinorHysteresis]	Defines the amount of fluctuation (hysteresis) from the Minor threshold, configured by the 'Minor Threshold (Yellow)' parameter in order for the threshold to be considered as crossed. The hysteresis is used only to determine threshold crossings to Green (i.e., from Yellow to Green, or Red to Green). In other words, the device considers a threshold crossing to Green only if the measured voice metric crosses the Minor threshold and the hysteresis. For example, if you configure the 'Minor Threshold (Yellow)' parameter to 4 and the 'Minor Hysteresis (Yellow)' parameter to

Parameter	Description
	0.1 (for MOS), the device considers a threshold crossing to Green only if the MOS crosses 4.1 (i.e., 4 + 0.1).
'Major Threshold (Red)' major-threshold-red [QOEColorRules_ MajorThreshold]	Defines the Major threshold value, which is the upper threshold located between the Yellow and Red states. To consider a threshold crossing: ■ **Increase in severity (i.e., Yellow to Red):** Only this value is used. ■ **Decrease in severity (Red to Yellow):** This value is used with the hysteresis, configured by the 'Major Hysteresis (Red)' parameter (see below). The valid threshold values are as follows: ■ MOS values are in multiples of 10. For example, to denote a MOS of 3.2, the value 32 (i.e., 3.2*10) must be entered. ■ Delay values are in msec. ■ Packet Loss values are in percentage (%). ■ Jitter is in msec. ■ Echo measures the Residual Echo Return Loss (RERL) in dB.
'Major Hysteresis (Red)' major-hysteresis-red [QOEColorRules_ MajorHysteresis]	Defines the amount of fluctuation (hysteresis) from the Major threshold, configured by the 'Major Threshold (Red)' parameter in order for the threshold to be considered as crossed. The hysteresis is used only to determine threshold crossings from Red to Yellow. In other words, the device considers a threshold crossing to Yellow only if the measured voice metric crosses the Major threshold and the hysteresis. For example, if you configure the 'Major Threshold (Red)' parameter to 2 and the 'Major Hysteresis (Red)' parameter to 0.1 (for MOS), the device considers a threshold crossing to Yellow only if the MOS crosses 2.1 (i.e., 2 + 0.1).

Configuring Bandwidth Profiles

The Bandwidth Profile table lets you configure up to 1,009 Bandwidth Profiles. A Bandwidth Profile defines bandwidth utilization thresholds for audio and/or video traffic (incoming and outgoing), which if crossed can result in various actions (discussed later in the section). Bandwidth Profiles enhance the device's monitoring of bandwidth utilization.

Once configured, you can apply Bandwidth Profiles to specific calls, by assigning them to any of the following configuration entities:

- IP Groups (see Configuring IP Groups)
- Media Realms (see Configuring Media Realms)
- Remote Media Subnets (see Configuring Remote Media Subnets)

Each time a configured bandwidth threshold is crossed, the device can do the following, depending on configuration:

- Reject calls destined to the IP Group or use an alternative IP Profile for the IP Group. For more information, see Configuring Quality of Service Rules.
- Use an alternative routing rule for alternative routing. If a call is rejected due to a crossed threshold, the device generates a SIP 806 response. You can configure the SIP response code as a reason for alternative routing (see Configuring SIP Response Codes for Alternative Routing Reasons).
- Send an SNMP alarm (acMediaRealmBWThresholdAlarm). The device clears the alarm when bandwidth utilization returns to normal (Green).

AudioCodes One Voice Operations Center (OVOC) displays bandwidth utilization using color-coded states:

- **Green:** Indicates bandwidth utilization is within normal range.
- **Yellow:** Indicates bandwidth utilization is encroaching on "total" bandwidth, serving as a warning (or it could also mean that bandwidth utilization has dropped below the red state).
- **Red:** Indicates that bandwidth utilization has exceeded total bandwidth.

Bandwidth Profiles let you configure bandwidth thresholds, which when crossed changes the color-coded state for bandwidth utilization:

- **Green-Yellow (Minor) Threshold:** Lower threshold configured as a percentage of the configured major (total) bandwidth threshold. When bandwidth goes over the threshold, the device considers it a Yellow state (Minor alarm severity); when it goes below the threshold, it considers it a Green state (cleared alarm).
- **Yellow-Red (Major) Threshold:** Upper threshold configured by the major (total) bandwidth threshold. When bandwidth goes over the threshold, the device considers it a Red state (Major alarm severity); when it goes below the threshold, it considers it a Yellow state (Minor alarm severity).

The device also uses hysteresis to determine whether the threshold has indeed being crossed. Hysteresis defines the amount of fluctuation from the threshold in order for the threshold to be considered as crossed (i.e., change in color state). Hysteresis is used to avoid false reports being sent by the device. Hysteresis is used only for threshold crossings toward a lesser severity (i.e., from Red to Yellow, Red to Green, or Yellow to Green). Hysteresis is configured as a percentage of the configured major (total) bandwidth threshold.

The following example is used to explain how the device considers threshold crossings. The example is based on a setup where the Major (total) bandwidth threshold is configured to 64,000 Kbps, the Minor threshold to 50% (of the total) and the hysteresis to 10% (of the total):

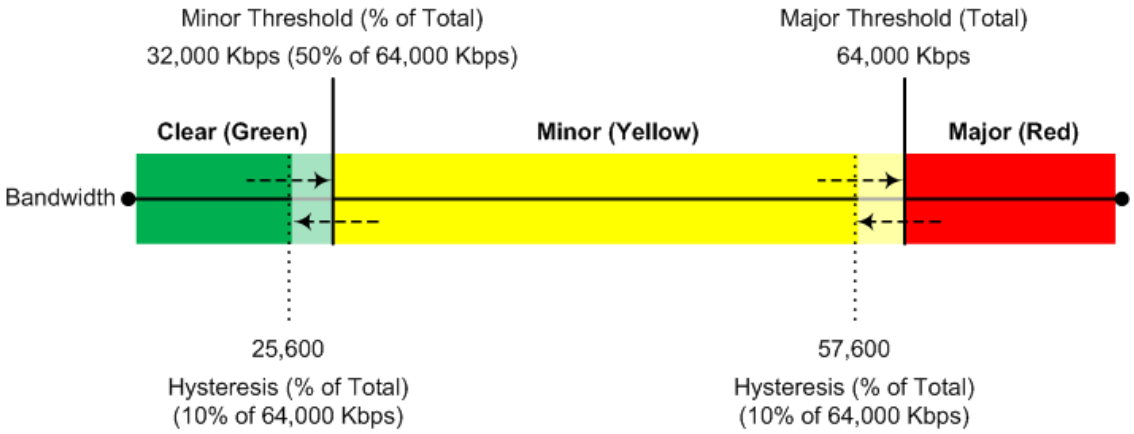


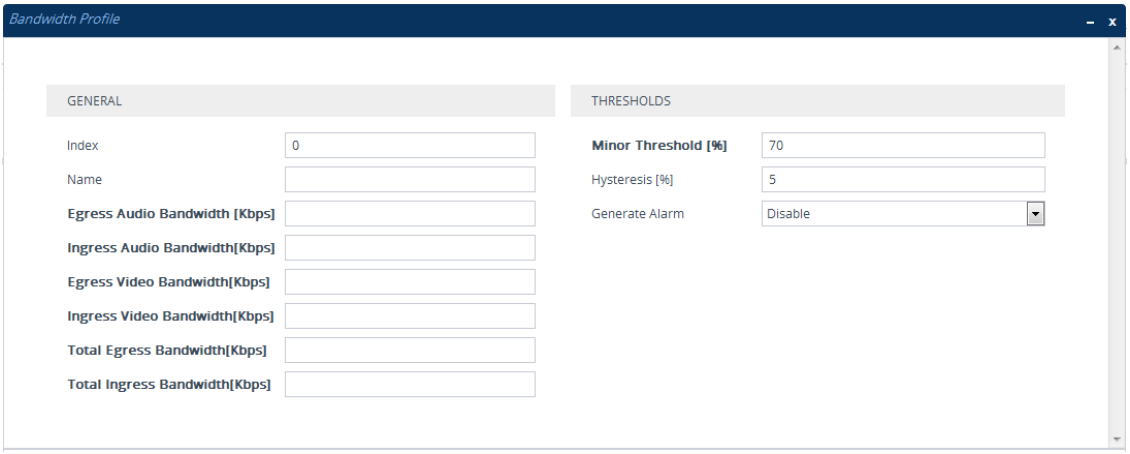
Table 17-5: Threshold Crossings based on Threshold and Hysteresis

Threshold Crossing	Calculation	Threshold based on Example
Green to Yellow (Minor alarm)	The change occurs if the current bandwidth crosses the configured Minor threshold only (i.e., hysteresis is not used).	32,000 Kbps
Green to Red (Major alarm)	The change occurs if the current bandwidth crosses the configured Major threshold only (i.e., hysteresis is not used).	64,000 Kbps
Yellow to Red (Major alarm)	The change occurs if the current bandwidth crosses the configured Major threshold only (i.e., hysteresis is not used).	64,000 Kbps
Red to Yellow (Minor alarm)	The change occurs if the current bandwidth crosses the configured Major threshold with hysteresis.	57,600 Kbps [64,000 - (10% x 64,000)]
Yellow to Green (alarm cleared)	The change occurs if the current bandwidth crosses the configured Minor threshold with hysteresis.	25,600 Kbps [32,000 - (10% x 64,000)]
Red to Green (alarm cleared)	The change occurs if the current bandwidth crosses the configured Minor threshold with hysteresis.	25,600 Kbps [32,000 - (10% x 64,000)]

The following procedure describes how to configure Bandwidth Profiles through the Web interface. You can also configure it through ini file [BWProfile] or CLI (`configure voip > qoe bw-profile`).

➢ To configure a Bandwidth Profile:

1. Open the Bandwidth Profile table (**Setup** menu > **Signaling & Media** tab > **Media** folder > **Quality of Experience** > **Bandwidth Profile**).
2. Click **New**; the following dialog box appears:



3. Configure a rule according to the parameters described in the table below.
4. Click **Apply**, and then reset the device with a save to flash memory.

Table 17-6: Bandwidth Profile Table Parameter Descriptions

Parameter	Description
General	
'Index' [BWProfile_Index]	Defines an index number for the new table row. **Note:** Each row must be configured with a unique index.
'Name' `name` [BWProfile_Name]	Defines a descriptive name, which is used when associating the row in other tables. The valid value is a string of up to 20 characters. **Note:** The parameter value cannot contain a forward slash (/).
'Egress Audio Bandwidth' `egress-audio-bandwidth` [BWProfile_ EgressAudioBandwidth]	Defines the major (total) threshold for outgoing audio traffic (in Kbps).
'Ingress Audio Bandwidth' `ingress-audio-bandwidth`	Defines the major (total) threshold for incoming audio traffic (in Kbps).

Parameter	Description
[BWProfile_ IngressAudioBandwidth]	
'Egress Video Bandwidth' `egress-video-bandwidth` [BWProfile_ EgressVideoBandwidth]	Defines the major (total) threshold for outgoing video traffic (in Kbps).
'Ingress Video Bandwidth' `ingress-video-bandwidth` [BWProfile_ IngressVideoBandwidth]	Defines the major (total) threshold for incoming video traffic (in Kbps).
'Total Egress Bandwidth' `total-egress-bandwidth` [BWProfile_ TotalEgressBandwidth]	Defines the major (total) threshold for video and audio outgoing bandwidth (in Kbps).
'Total Ingress Bandwidth' `total-ingress-bandwidth` [BWProfile_ TotalIngressBandwidth]	Defines the major (total) threshold for video and audio incoming bandwidth (in Kbps).
Thresholds	
'Minor Threshold' `minor-threshold` [BWProfile_MinorThreshold]	Defines the Minor threshold value, which is the lower threshold located between the Yellow and Green states. The parameter is configured as a percentage of the major (total) bandwidth threshold (configured by the above bandwidth parameters). For example, if you configure the parameter to 50 and the 'Egress Audio Bandwidth' parameter to 64,000, the Minor threshold for outgoing audio bandwidth is 32,000 (i.e., 50% of 64,000). To consider a threshold crossing: ■ **Increase in severity (i.e., Green to Yellow):** Only this value is used. ■ **Decrease in severity (Red to Green, or Yellow to Green):** This value is used with the hysteresis,

Parameter	Description
	configured by the 'Hysteresis' parameter (see below). **Note:** The parameter applies to all your configured bandwidths.
'Hysteresis' `hysteresis` [BWProfile_Hysteresis]	Defines the amount of fluctuation (hysteresis) from the configured bandwidth threshold in order for the threshold to be considered as crossed (i.e., avoids false reports of threshold crossings). The hysteresis is used only to determine threshold crossings when severity is reduced (i.e., from Red to Yellow, Yellow to Green, or Red to Green). The parameter is configured as a percentage of the Major (total) bandwidth threshold. For example, if you configure the parameter to 10 and the 'Egress Audio Bandwidth' parameter to 64,000, the hysteresis is 6,400 (10% of 64,000) and threshold crossings are considered at the following bandwidths: ■ Red-to-Yellow (Yellow-Minor alarm severity): 57,600 Kbps [64,000 - (10% x 64,000)] ■ Yellow-to-Green (Green-alarm cleared): 25,600 Kbps [32,000 - (10% x 64,000)]
'Generate Alarm' `generate-alarms` [BWProfile_GenerateAlarms]	Enables the device to send an SNMP alarm if a bandwidth threshold is crossed. ■ [0] **Disable** (default) ■ [1] **Enable**

Configuring Quality of Service Rules

The Quality of Service Rules table lets you configure up to 3,125 Quality of Service rules. A Quality of Service rule defines an action to perform when the threshold (major or minor) of a specific performance monitoring call metric is crossed for a specific IP Group. The call metric can be voice quality (i.e., MOS), bandwidth, Answer-seizure ratio (ASR), Network Effectiveness Ratio (NER), or Average Call Duration (ACD).

Depending on the call metric, you can configure the following actions to be performed if the threshold is crossed:

- Reject calls to the IP Group for a user-defined duration.

  Rejection of calls can also trigger alternative routing. When the device rejects a call due to an ASR, NER or ACD threshold crossing, it generates the SIP response code 850 (Signaling Limits Exceeded). When the device rejects a call due to Voice Quality and Bandwidth threshold crossing, it generates the SIP response code 806 (Media Limits Exceeded). If you

configure these SIP response codes for an Alternative Reasons Set (see Configuring SIP Response Codes for Alternative Routing Reasons) that is assigned to the IP Group ('SBC Alternative Routing Reasons Set' parameter) and the device rejects a call, it searches in the IP-to-IP Routing table for an alternative routing rule.

When the device rejects calls to an IP Group based on a Quality of Service rule, it raises an SNMP alarm (acIpGroupNoRouteAlarm). The alarm is also raised upon a keep-alive failure with the IP Group. For more information, refer to the *SNMP Reference Guide*.

- Use a different IP Profile for the IP Group or current call. This action can be useful, for example, when poor quality occurs due to packet loss and the device can then switch to an IP Profile configured with a higher RTP redundancy level or lower bit-rate coder.

To learn more about which actions are supported per call metric, see the description of the 'Rule Action' parameter below.

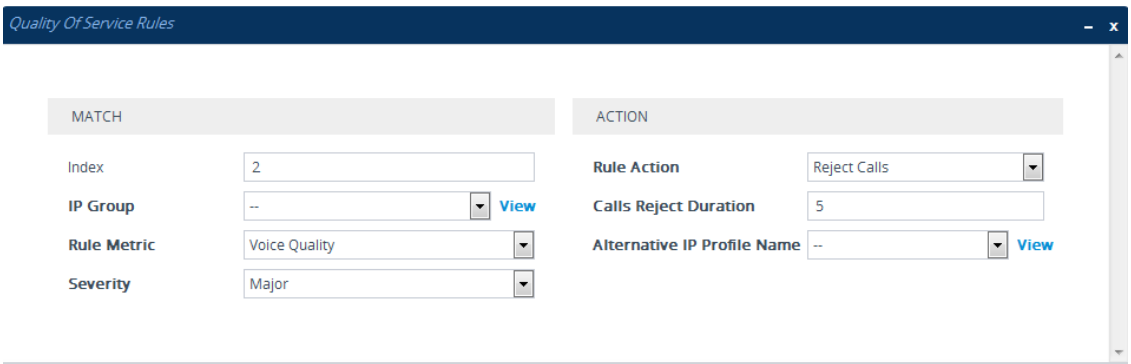
To configure thresholds, see the following sections:

- Voice Quality (MOS) - Configuring Quality of Experience Profiles
- Bandwidth - Configuring Bandwidth Profiles
- ASR, ACD and NER - Configuring Performance Profiles

The following procedure describes how to configure Quality of Service rules through the Web interface. You can also configure it through ini file [QualityOfServiceRules] or CLI (`configure voip > qoe quality-of-service-rules`).

➢ To configure a Quality of Service rule:

1. Open the Quality of Service Rules table (**Setup** menu > **Signaling & Media** tab > **Media** folder > **Quality of Service Rules**).
2. Click **New**; the following dialog box appears:



3. Configure a rule according to the parameters described in the table below.
4. Click **Apply**, and then reset the device with a save-to-flash for your settings to take effect.

Table 17-7: Quality of Service Rules Table Parameter Descriptions

Parameter	Description
Match	
'Index' [QualityOfServiceRules_Index]	Defines an index number for the new table row. **Note:** Each row must be configured with a unique index.
'IP Group' `ip-group-name` [QualityOfServiceRules_IPGroupName]	Assigns an IP Group. The rule applies to all calls belonging to the IP Group.
'Rule Metric' `rule-metric` [QualityOfServiceRules_RuleMetric]	Defines the performance monitoring call metric to which the rule applies if the metric's threshold is crossed. ■ [0] **Voice Quality** = (Default) The device calculates MOS of calls and if the threshold is crossed (i.e., poor quality), the configured action (see 'Rule Action' parameter below) is done for all **new** calls and for the **entire** IP Group. ■ [1] **Bandwidth** ■ [2] **ACD** ■ [3] **ASR** ■ [4] **NER** ■ [5] **Poor InVoice Quality** = The device calculates MOS (and TMMBR) of the call and if the threshold is crossed (i.e., poor quality), the device uses a different IP Profile (see 'Rule Action' parameter below) for the **current call only** (not the entire IP Group).
'Severity' `severity` [QualityOfServiceRules_Severity]	Defines the alarm severity level. When the configured severity occurs, the device performs the action of the rule. ■ [0] **Major** (Default) ■ [1] **Minor** **Note:** If you configure the 'Rule Metric' parameter to **ACD**, **ASR** or **NER**, you must configure the parameter to **Major**. For all other 'Rule Metric' parameter values, you can configure the parameter to any value.
Action	

Parameter	Description
'Rule Action' rule-action [QualityOfServiceRules_RuleAction]	Defines the action to be done if the rule is matched. ■ [0] **Reject Calls** = (Default) New calls destined to the specified IP Group are rejected for a user-defined duration. To configure the duration, use the 'Calls Reject Duration' parameter (see below). ■ [1] **Alternative IP Profile** = A different IP Profile is used for the IP Group or call (depending on the 'Rule Metric' parameter). To specify the IP Profile, use the 'Alternative IP Profile Name' parameter (see below). **Note:** ■ If you configure the 'Rule Metric' parameter to **ACD**, **ASR** or **NER**, you must configure the parameter to **Reject Calls**. ■ If you configure the 'Rule Metric' parameter to **Voice Quality** or **Bandwidth**: ✓ If you configure the 'Severity' parameter to **Minor**, you must configure the parameter to **Alternative IP Profile**. ✓ If you configure the 'Severity' parameter to **Major**, you can configure the parameter to any option. When configured to **Alternative IP Profile** and the threshold is crossed, the device changes the IP Profile for the entire IP Group for all new calls. ■ If you configure the 'Rule Metric' parameter to **Poor InVoice Quality**, you must configure the parameter to **Alternative IP Profile**. If the threshold is crossed (i.e., poor call quality), the device changes the IP Profile for the specific call **only** (during the call).
'Calls Reject Duration' calls-reject-duration [QualityOfServiceRules_CallsRejectDuration]	Defines the duration (in minutes) for which the device rejects calls to the IP Group if the rule is matched. The default is 5. **Note:** The parameter is applicable only if the 'Rule Action' parameter is configured to **Reject Calls**.
'Alternative IP Profile Name' alt-ip-profile-name	Assigns a different IP Profile to the IP Group or call (depending on the 'Rule Metric' parameter) if the rule is matched. By default, no value is defined.

Parameter	Description
[QualityOfServiceRules_ AltIPProfileName]	**Note:** The parameter is applicable only if the 'Rule Action' parameter is configured to **Alternative IP Profile**.

18 Core Entities

This section describes configuration of core SIP entities.

Configuring Media Realms

The Media Realms table lets you configure a pool of up to 1,024 SIP media interfaces, termed *Media Realms*. Media Realms lets you divide a Media-type interface (configured in the IP Interfaces table) into several media realms, where each realm is specified by a UDP port range. Media Realms also define the maximum number of permitted media sessions.

Once configured, to apply Media Realms to specific calls, you need to assign them to any of the following configuration entities:

- IP Groups (see Configuring IP Groups)
- SIP Interfaces (see Configuring SIP Interfaces)

You can also apply the device's Quality of Experience feature to Media Realms:

- **Quality of Experience Profile:** Call quality monitoring based on thresholds for voice metrics (e.g., MOS) can be applied per Media Realm. For example, if MOS is considered poor, calls on this Media Realm can be rejected. To configure Quality of Experience Profiles, see Configuring Quality of Experience Profiles.
- **Bandwidth Profile:** Bandwidth utilization thresholds can be applied per Media Realm. For example, if bandwidth thresholds are crossed, the device can reject any new new calls on this Media Realm. To configure Bandwidth Profiles, see Configuring Bandwidth Profiles.

The Media Realms table provides the following "child" tables:

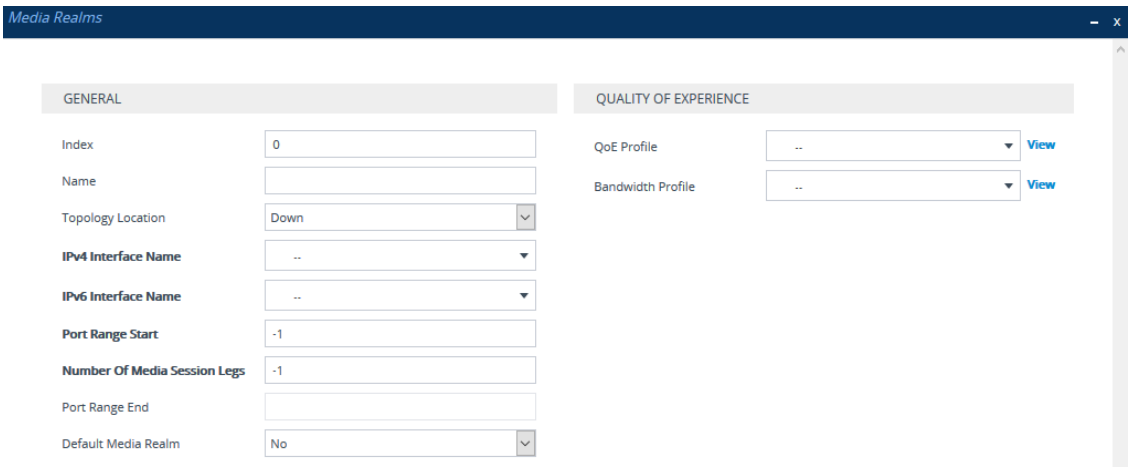
- Remote Media Subnets: Defines remote destination subnets per Media Realm and assigns each subnet a Quality of Experience Profile and Bandwidth Profile. For more information, see Configuring Remote Media Subnets.
- Media Realm Extensions: Defines port ranges for multiple Media-type interfaces per Media Realm. For more information, see Configuring Media Realm Extensions.

- The Media Realm assigned to an IP Group overrides any other Media Realm assigned to any other configuration entity associated with the call.
- If you modify a Media Realm that is currently being used by a call, the device does not perform Quality of Experience for the call.
- If you delete a Media Realm that is currently being used by a call, the device maintains the call until the call parties end the call.
- The device provides a default Media Realm ("DefaultRealm"), which you can modify or delete.

The following procedure describes how to configure Media Realms through the Web interface. You can also configure it through ini file [CpMediaRealm] or CLI (`configure voip > realm`).

➢ To configure a Media Realm:

1. Open the Media Realms table (**Setup** menu > **Signaling & Media** tab > **Core Entities** folder > **Media Realms**).
2. Click **New**; the following dialog box appears:



3. Configure the Media Realm according to the parameters described in the table below.
4. Click **Apply**.

Table 18-1: Media Realms table Parameter Descriptions

Parameter	Description
General	
'Index' [CpMediaRealm_Index]	Defines an index number for the new table row. **Note:** Each row must be configured with a unique index.
'Name' `name` [CpMediaRealm_ MediaRealmName]	Defines a descriptive name, which is used when associating the row in other tables. The valid value is a string of up to 39 characters. **Note:** ■ The parameter is mandatory. ■ Each row must be configured with a unique name. ■ The parameter value cannot contain a forward slash (/).
'Topology Location' `topology-location` [CpMediaRealm_ TopologyLocation]	Defines the display location of the Media Realm in the Topology view. ■ [0] **Down** = (Default) The Media Realm element is displayed on the lower border of the view.

Parameter	Description
	■ [1] **Up** = The Media Realm element is displayed on the upper border of the view. For more information on the Topology view, see Building and Viewing SIP Entities in Topology View.
'IPv4 Interface Name' `ipv4` [CpMediaRealm_IPv4IF]	Assigns an IPv4 network interface to the Media Realm. By default, no value is defined. To configure IP network interfaces, see Configuring IP Network Interfaces.
'IPv6 Interface Name' `ipv6if` [CpMediaRealm_IPv6IF]	Assigns an IPv6 network interface to the Media Realm. By default, no value is defined. To configure IP network interfaces, see Configuring IP Network Interfaces.
'UDP Port Range Start' `port-range-start` [CpMediaRealm_PortRangeStart]	Defines the starting port for the range of media interface UDP ports. By default, no value is defined. **Note:** ■ You must configure **all** your Media Realms with port ranges or all without; not some with and some without. ■ The available UDP port range is according to the [BaseUDPport] parameter. For more information, see Configuring RTP Base UDP Port. ■ The port number **must** be different from ports configured for SIP traffic (i.e., ports configured for SIP Interfaces) that use the same IP Interface. For example, if the RTP port range is 6000 to 6999, the SIP port can be less than 6000 or greater than 6999. ■ Media Realms associated with the same IP Interface must not have overlapping port ranges. ■ Media Realms and Media Realm Extensions associated with the same IP Interface must not have overlapping port ranges.
'Number of Media' Session Legs	Defines the number of media sessions for the

Parameter	Description
`session-leg` [CpMediaRealm_ MediaSessionLeg]	configured port range. By default, no value is defined.

'UDP Port Range End'
`port-range-end`
[CpMediaRealm_PortRangeEnd]
	(Read-only field) Displays the ending port for the range of media interface UDP ports. The device automatically populates the parameter with a value, calculated by the summation of the 'UDP Port Range Start' parameter and 'Number of Media Session Legs' parameter (multiplied by the port chunk size) minus 1:
	`start port + (sessions * port spacing) - 1`
	For example, a port starting at 6,000, 5 sessions and 10 port spacing:
	`6,000 + (5 * 10) - 1 = 6,000 + (50) - 1 = 6,000 + 49 = 6,049`
	The device allocates the UDP ports for RTP, RTCP and T.38 traffic per leg in "jumps" (spacing) of 4, 5 or 10, configured by the UdpPortSpacing parameter. For example, if the port range starts at 6000 and the UDP port spacing is 10, the available ports include 6000, 6010, 6020, 6030, and so on (depending on number of media sessions).
	For RTCP and T.38 traffic, the port offset from the RTP port used for the voice session is one and two, respectively. For example, if the voice session uses RTP port 6000, the RTCP port and T.38 port for the session is 6001 and 6002, respectively. However, you can configure the device to use the same port for RTP and T.38 packets, by configuring the [T38UseRTPPort] parameter to [1].
	For more information on local UDP port range, see Configuring RTP Base UDP Port.

'TCP Port Range Start' `tcp-port-range-start` [CpMediaRealm_ TCPPortRangeStart]	Defines the starting port of the range of TCP ports for MSRP traffic. The device allocates the ports consecutively to traffic. For example, if the port range starts at 5000 and ends at 5100, the device first allocates port 5000, then 5001, then 5002, and so on. The valid value is 4000 to 32768. The default is 0. For MSRP, the port number is used in the SDP's

Parameter	Description
	'a=path' line. For more information on MSRP, see Configuring Message Session Relay Protocol on page 779. **Note:** ■ Make sure that you also configure the ending port (see the 'TCP Port Range End' parameter, below). ■ Media Realms associated with the same IP Interface must not have overlapping port ranges. ■ MSRP ports do not support Media Realm Extensions.
'TCP Port Range End' `tcp-port-range-end` [CpMediaRealm_ TCPPortRangeEnd]	Defines the ending port of the range of TCP ports for MSRP traffic. The device allocates the ports consecutively to traffic. For example, if the port range starts at 5000 and ends at 5100, the device first allocates port 5000, then 5001, then 5002, and so on. The valid value is 4000 to 32768. The default is 0. For MSRP, the port number is used in the SDP's 'a=path' line. For more information on MSRP, see Configuring Message Session Relay Protocol on page 779. **Note:** ■ Make sure that you also configure the starting port (see the 'TCP Port Range Start' parameter, above). ■ Media Realms associated with the same IP Interface must not have overlapping ports. ■ The port range cannot overlap with TCP ports configured for SIP traffic (i.e., SIP Interfaces) that use the same IP Interface. For example, if the TCP port range is 6000 to 6999, the SIP Interface's TCP port must be less than 6000 or greater than 6999. ■ MSRP ports do not support Media Realm Extensions.
'Default Media Realm' `is-default` [CpMediaRealm_IsDefault]	Defines the Media Realm as the default Media Realm. The default Media Realm is used for SIP Interfaces and IP Groups for which you have not assigned a Media Realm.

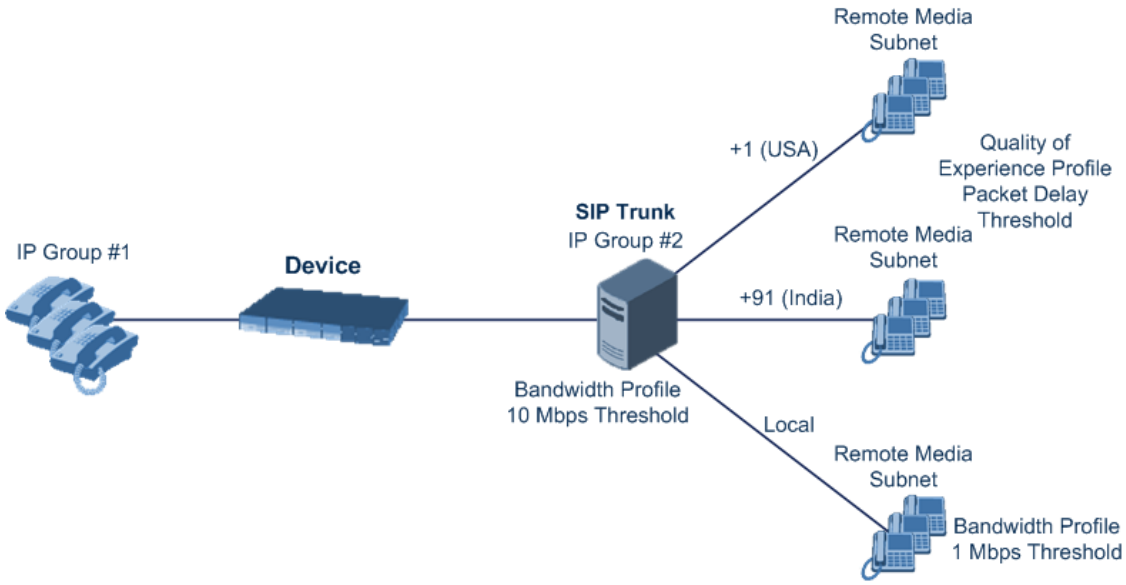
Parameter	Description
	■ [0] **No** (default) ■ [1] **Yes** **Note:** ■ You can configure the parameter to **Yes** for only **one** Media Realm; all the other Media Realms must be configured to **No**. ■ If you do not configure the parameter (i.e., the parameter is **No** for all Media Realms), the device uses the first Media Realm in the table as the default. ■ If the table is not configured, the default Media Realm includes all configured media interfaces.
'Used By Routing Server' `used-by-routing-server` [CpMediaRealm_ UsedByRoutingServer]	Enables the Media Realm to be used by a third-party routing server or ARM for call routing decisions. ■ [0] **Not Used** (default) ■ [1] **Used** For more information on the third-party routing server or ARM feature, see Centralized Third-Party Routing Server.
Quality of Experience	
'QoE Profile' `qoe-profile` [CpMediaRealm_QoeProfile]	Assigns a QoE Profile to the Media Realm. By default, no value is defined. To configure QoE Profiles, see Configuring Quality of Experience Profiles.
'BW Profile' `bw-profile` [CpMediaRealm_BWProfile]	Assigns a Bandwidth Profile to the Media Realm. By default, no value is defined. To configure Bandwidth Profiles, see Configuring Bandwidth Profiles.

Configuring Remote Media Subnets

Remote Media Subnets define destination subnets for media (RTP/SRTP) traffic on a specific Media Realm. Each Remote Media Subnet can be assigned different call quality (Quality of Experience Profile) and bandwidth utilization (Bandwidth Profile) profiles. These profiles are configured in Configuring Quality of Experience Profiles and Configuring Bandwidth Profiles,

respectively. Thus, you can apply these profiles to remote media subnets instead of Media Realms or IP Groups. You can configure up to five Remote Media Subnets per Media Realm.

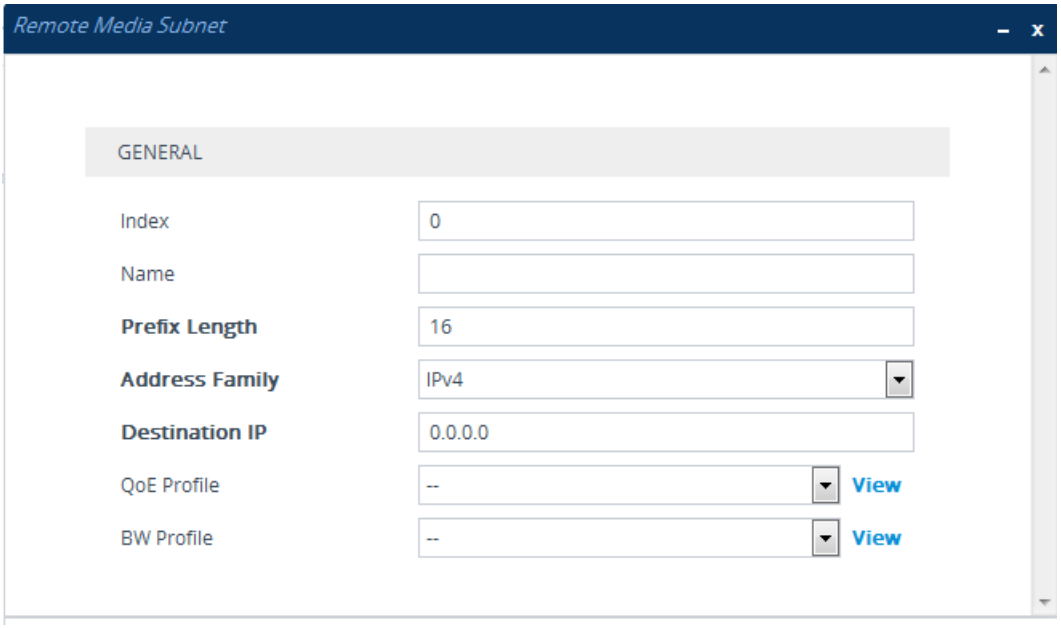
The figure below illustrates an example for implementing Remote Media Subnets. IP Group #2 represents a SIP Trunk which routes international (USA and India) and local calls. As international calls are typically more prone to higher delay than local calls, different Quality of Experience Profiles are assigned to them. This is done by creating Remote Media Subnets for each of these call destinations and assigning each Remote Media Subnet a different Quality of Experience Profile. A Quality of Experience Profile that defines a packet delay threshold is assigned to the international calls, which if crossed, a different IP Profile is used that defines higher traffic priority to voice over other traffic. In addition, IP Group #2 has a 10-Mbps bandwidth threshold and a "tighter" bandwidth limitation (e.g., 1 Mbps) is allocated to local calls. If this limit is exceeded, the device rejects new calls to this Remote Media Subnet.



The following procedure describes how to configure Remote Media Subnets through the Web interface. You can also configure it through ini file [RemoteMediaSubnet] or CLI (`configure voip > remote-media-subnet`).

➢ To configure a Remote Media Subnet:

1. Open the Media Realms table (see Configuring Media Realms).
2. Select the Media Realm row for which you want to add Remote Media Subnets, and then click the **Remote Media Subnet** link located below the table; the Remote Media Subnet table appears.
3. Click **New**; the following dialog box appears:



4. Configure the Remote Media Subnet according to the parameters described in the table below.
5. Click **Apply**.

Table 18-2: Remote Media Subnet Table Parameter Descriptions

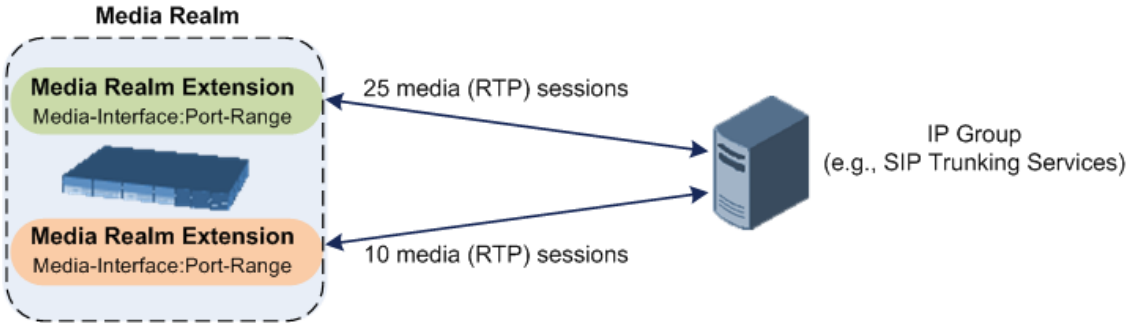
Parameter	Description
'Index' [RemoteMediaSubnet_ RemoteMediaSubnetIndex]	Defines an index number for the new table row. **Note:** Each row must be configured with a unique index.
'Name' `name` [RemoteMediaSubnet_ RemoteMediaSubnetName]	Defines a descriptive name, which is used when associating the row in other tables. The valid value is a string of up to 20 characters. **Note:** Each row must be configured with a unique name.
'Prefix Length' `prefix-length` [RemoteMediaSubnet_ PrefixLength]	Defines the subnet mask in Classless Inter-Domain Routing (CIDR) notation. For example, 16 denotes 255.255.0.0. The default is 16.
'Address Family' `address-family` [RemoteMediaSubnet_ AddressFamily]	Defines the IP address protocol. ■ [2] **IPv4** (default) ■ [10] **IPv6**
'Destination IP'	Defines the IP address of the destination.

Parameter	Description
`dst-ip-address` [RemoteMediaSubnet_DstIPAddress]	The default is 0.0.0.0.
'QoE Profile' `qoe-profile` [RemoteMediaSubnet_QOEProfileName]	Assigns a Quality of Experience Profile to the Remote Media Subnet. By default, no value is defined. To configure QoE Profiles, see Configuring Quality of Experience Profiles.
'BW Profile' `bw-profile` [RemoteMediaSubnet_BWProfileName]	Assigns a Bandwidth Profile to the Remote Media Subnet. By default, no value is defined. To configure Bandwidth Profiles, see Configuring Bandwidth Profiles.

Configuring Media Realm Extensions

The Media Realm Extension table lets you configure up to 5,120 Media Realm Extensions. A Media Realm Extension is associated with a specific Media Realm and defines a port range and the number of media sessions for a specific Media-type network interface (configured in the IP Interfaces table). Therefore, a Media Realm Extension enhances a Media Realm by allowing you to define different port ranges, media sessions, and network interface than is defined by the associated Media Realm (i.e., the Media Realm is distributed across multiple interfaces).

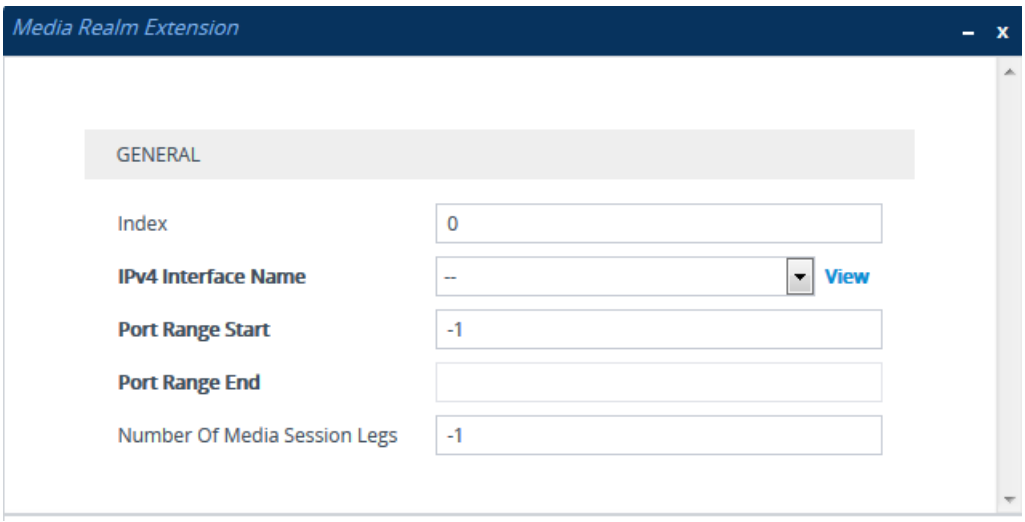
Media Realm Extensions can be useful, for example, to overcome limitations of the maximum number of media ports supported per interface. Instead of configuring only a single Media Realm in the Media Realms table (see Configuring Media Realms), you can also configure additional "Media Realms" in the Media Realm Extensions table associated with the single Media Realm. An IP Group that is associated with a Media Realm configured with Media Realm Extensions, allocates its media sessions / ports between the different interfaces, as configured by the Media Real and its associated Media Realm Extensions. For example, two Media Realm Extensions could be configured, whereby one allocates 25 media sessions on interface "LAN-1" and another, 10 sessions on interface "LAN-2". The Media Realm associated with these Media Realm Extensions would be assigned to the relevant IP Group.



The following procedure describes how to configure Media Realm Extensions through the Web interface. You can also configure it through ini file [MediaRealmExtension] or CLI (`configure voip > voip-network realm-extension`).

➢ **To configure a Media Realm Extension:**

1. Open the Media Realms table (see Configuring Media Realms).
2. Select the Media Realm for which you want to add Remote Media Extensions, and then click the **Media Realm Extension** link located below the table; the Media Realm Extension table appears.
3. Click **New**; the following dialog box appears:



4. Configure the Media Realm Extension according to the parameters described in the table below.
5. Click **Apply**.

Table 18-3: Media Realm Extension Table Parameter Descriptions

Parameter	Description
'Index' [MediaRealmExtension_ ExtensionIndex]	Defines an index number for the new table row. **Note:** Each row must be configured with a unique index.
'IPv4 Interface Name' [MediaRealmExtension_ IPv4IF]	Assigns an IPv4 network interface (configured in the IP Interfaces table) to the Media Realm Extension. By default, no value is defined. To configure IP network interfaces, see Configuring IP Network Interfaces. **Note:** ■ The parameter is mandatory.

Parameter	Description
	■ You must configure the Media Realm Extension with an IP network interface that has the same IP version(s) as the Media Realm to which the Media Realm Extension is associated. If the associated Media Realm is assigned both an IPv4 and IPv6 network interface, you also need to assign the Media Realm Extension with both an IPv4 and IPv6 network interface. For example, if the associated Media Realm is assigned only an IPv4 network interface, you also need to assign the Media Realm Extension with an IPv4 network interface.
'IPv6 Interface Name' [MediaRealmExtension_IPv6IF]	Assigns an IPv6 network interface (configured in the IP Interfaces table) to the Media Realm Extension. By default, no value is defined. **Note:** ■ The parameter is mandatory. ■ You must configure the Media Realm Extension with an IP network interface that has the same IP version(s) as the Media Realm to which the Media Realm Extension is associated. If the associated Media Realm is assigned both an IPv4 and IPv6 network interface, you also need to assign the Media Realm Extension with both an IPv4 and IPv6 network interface. For example, if the associated Media Realm is assigned an IPv6 network interface, you also need to assign the Media Realm Extension with an IPv6 network interface.
'Port Range Start' [MediaRealmExtension_PortRangeStart]	Defines the first (lower) port in the range of media UDP ports for the Media Realm Extension. By default, no value is defined. **Notes:** ■ You must either configure all your Media Realms with port ranges or all without; not some with and some without. ■ The available UDP port range is according to the [BaseUDPport] parameter (see Configuring RTP Base UDP Port). ■ The port range must not overlap with any other media port range configured for other Media Realm Extensions, Media Realms, or SIP Interfaces that are associated with

Parameter	Description
	the same IP network interface.
'Port Range End' [MediaRealmExtension_ PortRangeEnd]	Defines the last (upper) port in the range of media UDP ports for the Media Realm Extension. **Note:** It is unnecessary to configure the parameter. The device automatically populates the parameter with a value, calculated by the summation of the 'Number of Media Session Legs' parameter (multiplied by the port chunk size) and the 'Port Range Start' parameter. After you have added the Media Realm Extension row to the table, the parameter is displayed with the calculated value.
'Number Of Media Session Legs' [MediaRealmExtension_ MediaSessionLeg]	Defines the number of media sessions for the port range. For example, 100 ports correspond to 10 media sessions, since ports are allocated in chunks of 10. By default, no value is defined. **Note:** The parameter is mandatory.

Configuring SRDs

The SRDs table lets you configure up to 280 signaling routing domains (SRD). The SRD is a logical representation of an entire SIP-based VoIP network (Layer 5) consisting of groups of SIP users and servers. The SRD is associated with all the configuration entities (e.g., SIP Interfaces and IP Groups) required for routing calls within the network. Typically, only a **single** SRD is required (recommended) for most deployments. Multiple SRDs are only required for multi-tenant deployments, where the physical device is "split" into multiple logical devices. For more information on multi-tenant architecture, see Multiple SRDs for Multi-tenant Deployments.

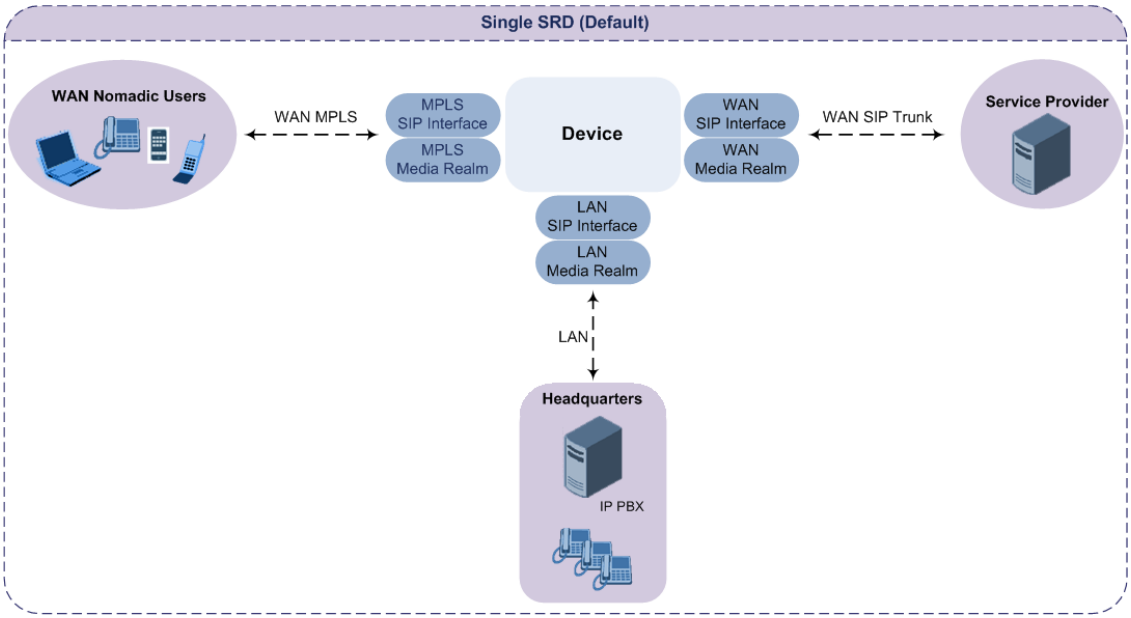
As the device is shipped with a default SRD ("DefaultSRD" at Index 0), if your deployment requires only one SRD, you can use the default SRD instead of creating a new one. When only one SRD is employed and you create other related configuration entities (e.g., SIP Interfaces), the default SRD is automatically assigned to the new configuration entity. Therefore, when employing a single-SRD configuration topology, there is no need to handle SRD configuration (i.e., transparent).

You can assign SRDs to the following configuration entities:

- SIP Interface (mandatory) - see Configuring SIP Interfaces
- IP Group (mandatory) - see Configuring IP Groups
- Proxy Set (mandatory) - see Configuring Proxy Sets
- (SBC application only) Classification rule - see Configuring Classification Rules

As mentioned previously, if you use only a single SRD, the device automatically assigns it to the above-listed configuration entities.

As each SIP Interface defines a different Layer-3 network (see Configuring SIP Interfaces for more information) on which to route or receive calls and as you can assign multiple SIP Interfaces to the same SRD, for most deployment scenarios (even for multiple Layer-3 network environments), you only need to employ a single SRD to represent your VoIP network (Layer 5). For example, if your VoIP deployment consists of an corporate IP PBX (LAN), a SIP Trunk (WAN), and far-end users (WAN), you would only need a single SRD. The single SRD would be assigned to three different SIP Interfaces, where each SIP Interface would represent a specific Layer-3 network (IP PBX, SIP Trunk, or far-end users) in your environment. The following figure provides an example of such a deployment:

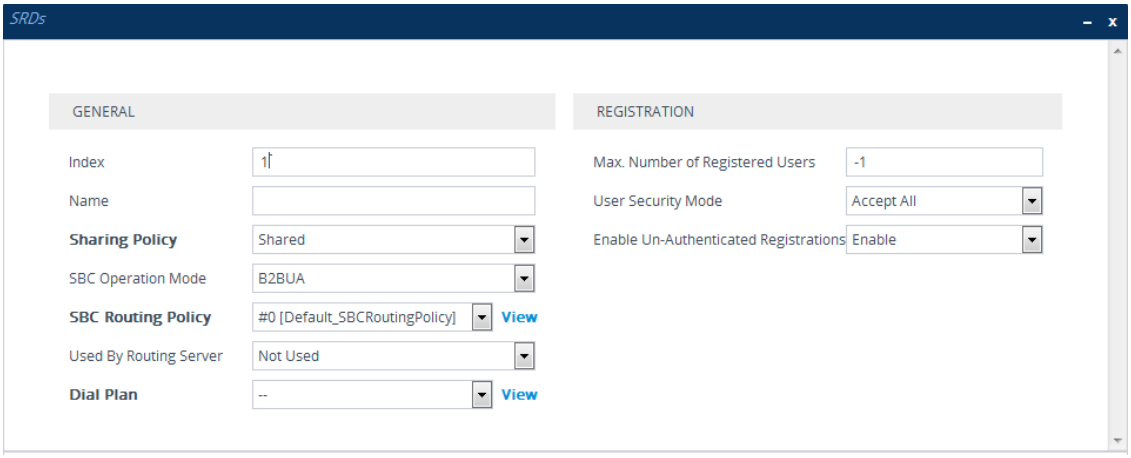


- It is recommended to use a single-SRD configuration topology, unless you are deploying the device in a multi-tenant environment, in which case multiple SRDs are required.
- Each SIP Interface, Proxy Set, and IP Group can be associated with only one SRD.
- If you have upgraded your device to Version 7.0 and your device was configured with multiple SRDs but not operating in a multi-tenant environment, it is recommended to gradually change your configuration to a single SRD topology.
- If you upgrade the device from an earlier release to Version 7.0, your previous SRD configuration is fully preserved regarding functionality. The same number of SRDs is maintained, but the configuration elements are changed to reflect the configuration topology of Version 7.0. Below are the main changes in configuration topology when upgrading to Version 7.0:
  - ✓ The SIP Interface replaces the associated SRD in several tables (due to support for multiple SIP Interfaces per SRD).
  - ✓ Some fields in the SRDs table were duplicated or moved to the SIP Interfaces table.
  - ✓ Indices used for associating configuration entities in tables are changed to row pointers (using the entity's name).
  - ✓ Some tables are now associated (mandatory) with an SRD (SIP Interface, IP Group, Proxy Set, and Classification).
  - ✓ Some fields used for associating configuration entities in tables now have a value of **Any** to distinguish between **Any** and **None** (deleted entity or not associated).

The following procedure describes how to configure SRDs through the Web interface. You can also configure it through ini file [SRD] or CLI (`configure voip > srd`).

➢ To configure an SRD:

1. Open the SRDs table (**Setup** menu > **Signaling & Media** tab > **Core Entities** folder > **SRDs**).
2. Click **New**; the following dialog box appears:



3. Configure an SRD according to the parameters described in the table below.
4. Click **Apply**.

Table 18-4: SRDs table Parameter Descriptions

Parameter	Description
General	
'Index' [SRD_Index]	Defines an index for the new table row. **Note:** Each row must be configured with a unique index.
'Name' `name` [SRD_Name]	Defines a descriptive name, which is used when associating the row in other tables. The valid value can be a string of up to 40 characters. **Note:** ■ The parameter is mandatory. ■ Each row must be configured with a unique name. ■ The parameter value cannot contain a forward slash (/).
'Sharing Policy' `type` [SRD_SharingPolicy]	Defines the sharing policy of the SRD, which determines whether the SRD shares its SIP resources (SIP Interfaces, Proxy Sets, and IP Groups) with all other SRDs (Shared and Isolated). ■ [0] **Shared** = (Default) SRD shares its resources with other SRDs (Isolated and Shared) and calls can thus be routed between the SRD and other SRDs. ■ [1] **Isolated** = SRD does not share its resources with other SRDs and calls cannot be routed between the SRD and other Isolated SRDs. However, calls can be routed between the SRD and other Shared SRDs. For more information on SRD Sharing Policy, see Multiple SRDs for Multi-tenant Deployments.
'SBC Operation Mode' `sbc-operation-mode` [SRD_SBCOperationMode]	Defines the device's operational mode for the SRD. ■ [0] **B2BUA** = (Default) Device operates as a back-to-back user agent (B2BUA), changing

Parameter	Description
	the call identifiers and headers between the inbound and outbound legs. ■ [1] **Call Stateful Proxy** = Device operates as a Stateful Proxy, passing the SIP message transparently between inbound and outbound legs. In other words, the same SIP dialog identifiers (tags, Call-Id and CSeq) occur on both legs (as long as no other configuration disrupts the CSeq compatibleness). For more information on B2BUA and Stateful Proxy modes, see B2BUA and Stateful Proxy Operating Modes. **Note:** ■ The settings of the parameter also determines the default behavior of related parameters in the IP Profiles table (SBCRemoteRepresentationMode, SBCKeepVIAHeaders, SBCKeepUserAgentHeader, SBCKeepRoutingHeaders, SBCRemoteMultipleEarlyDialogs). ■ If the 'SBC Operation Mode' parameter is configured in the IP Groups table, the 'SBC Operation Mode' parameter in the SRDs table is ignored.
'SBC Routing Policy' `sbc-routing-policy-name` [SRD_SBCRoutingPolicyName]	Assigns a Routing Policy to the SRD. By default, no value is defined if you have configured multiple Routing Policies. If you have configured only one Routing Policy, the device assigns it to the SRD by default. For more information on Routing Policies, see Configuring SBC Routing Policy Rules. **Note:** ■ If you have assigned a Routing Policy to a Classification rule that is associated with the SRD, the Routing Policy assigned to the SRD is ignored. ■ You can assign the same Routing Policy to

Parameter	Description
	multiple SRDs.
'Used By Routing Server' `used-by-routing-server` [SRD_UsedByRoutingServer]	Enables the SRD to be used by a third-party routing server for call routing decisions. ■ [0] **Not Used** (default) ■ [1] **Used** For more information on the third-party routing server feature, see Centralized Third-Party Routing Server.
'Dial Plan' `sbc-dial-plan-name` [SRD_SBCDialPlanName]	Assigns a Dial Plan to the SRD. The device searches the Dial Plan for a dial plan rule that matches the source number and if not found, for a rule that matches the destination number. If a matching dial plan rule is found, the rule's tag is used in the routing and/or manipulation processes as source and/or destination tags. To configure Dial Plans, see Configuring Dial Plans.
'CAC Profile' `cac-profile` [SRD_AdmissionProfile]	Assigns a Call Admission Control Profile (CAC rules) to the SRD. By default, no value is defined. To configure CAC Profiles, see Configuring Call Admission Control on page 696.
Registration	
'Max. Number of Registered Users' `max-reg-users` [SRD_MaxNumOfRegUsers]	Defines the maximum number of users belonging to the SRD that can register with the device. The default is -1, which means that the number of allowed user registrations is unlimited.
'User Security Mode' `block-un-reg-users` [SRD_BlockUnRegUsers]	Defines the blocking (reject) policy for incoming SIP dialog-initiating requests (e.g., INVITE messages) from registered and unregistered users belonging to the SRD. ■ [0] **Accept All** = (Default) Accepts requests from registered and unregistered users. ■ [1] **Accept Registered Users** = Accepts

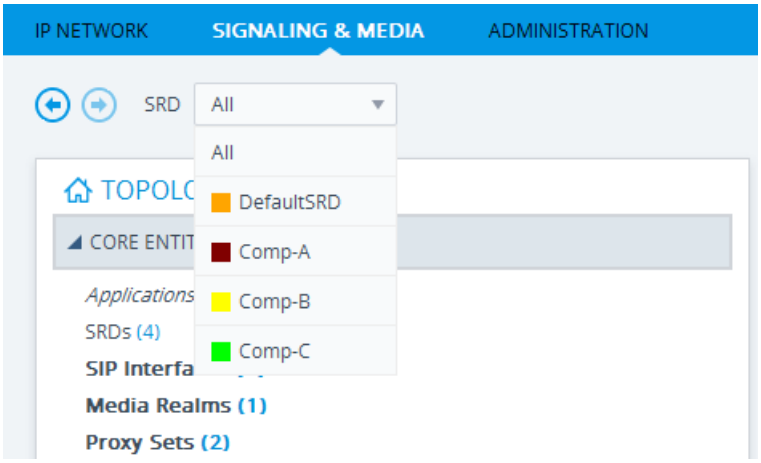
Parameter	Description
	requests only from users registered with the device. Requests from users not registered are rejected. ■ [2] **Accept Registered Users from Same Source** = Accepts requests only from registered users whose source address is the same as that registered with the device (during the REGISTER message process). All other requests are rejected. If the transport protocol is UDP, the verifies the IP address and port; otherwise, it verifies only the IP address. The verification is performed before any of the device's call handling processes (i.e., Classification, Manipulation and Routing). **Note:** ■ The parameter is applicable only to calls belonging to User-type IP Groups. ■ The feature is not applicable to REGISTER requests. ■ The option, Accept Registered Users from Same Source [2] does not apply to registration refreshes. These requests are accepted even if the source address is different to that registered with the device. ■ When the device rejects a call, it sends a SIP 500 "Server Internal Error" response to the user. In addition, it reports the rejection (Dialog establish failure - Classification failure) using the Intrusion Detection System (IDS) feature (see Configuring IDS Policies), by sending an SNMP trap. ■ When the corresponding parameter in the SIP Interfaces table (SIPInterface_ BlockUnRegUsers) is configured to any value other than default [-1] for a SIP Interface that is associated with the SRD, the parameter in the SRDs table is ignored for calls belonging to the SIP Interface.

Parameter	Description
'Enable Un-Authenticated Registrations' `enable-un-auth-registrs` [SRD_ EnableUnAuthenticatedRegistrations]	Enables the device to accept REGISTER requests and register them in its registration database from new users that have not been authenticated by a proxy/registrar server (due to proxy down) and thus, re-routed to a User-type IP Group. In normal operation scenarios in which the proxy server is available, the device forwards the REGISTER request to the proxy and if authenticated by the proxy (i.e., device receives a success response), the device adds the user to its registration database. The routing to the proxy is according to the SBC IP-to-IP Routing table where the destination is the proxy's IP Group. However, when the proxy is unavailable (e.g., due to network connectivity loss), the device can accept REGISTER requests from new users if a matching alternative routing rule exists in the SBC IP-to-IP Routing table where the destination is the user's User-type IP Group (i.e., call survivability scenarios) and if the parameter is enabled. ■ [0] **Disable** = The device rejects REGISTER requests from new users that were not authenticated by a proxy server. ■ [1] **Enable** = (Default) The device accepts REGISTER requests from new users even if they were not authenticated by a proxy server, and registers the user in its registration database. **Note:** ■ Regardless of the parameter, the device always accepts registration refreshes from users that are already registered in its database. ■ For a SIP Interface that is associated with the SRD, if the corresponding parameter in the SIP Interfaces table (SIPInterface_ EnableUnAuthenticatedRegistrations) is configured to Disable or Enable, the

Parameter	Description
	parameter in the SRD is ignored for calls belonging to the SIP Interface.

Filtering Tables in Web Interface by SRD

When your configuration includes multiple SRDs, you can filter tables in the Web interface by SRD. The filter is configured in the SRD Filter drop-down list, located on the Web interface's toolbar, as shown below.



The filter is applied throughout the Web GUI. When you select an SRD for filtering, the Web interface displays only table rows associated with the filtered SRD. When you add a new row to a table, the filtered SRD is automatically selected as the associated SRD. For example, if you filter the Web display by SRD "Comp-A" and you then add a new Proxy Set, the Proxy Set is automatically associated with this SRD (i.e., the 'SRD' parameter is set to "Comp-A"). All other parameters in the dialog box are also automatically set to values associated with the filtered SRD.

The SRD filter also affects display of number of configured rows and invalid rows by status icons on table items in the Navigation tree. The status icons only display information relating to the filtered SRD.

SRD filtering is especially useful in multi-tenant setups where multiple SRDs may be configured. In such a setup, SRD filtering eliminates configuration clutter by "hiding" SRDs that are irrelevant to the current configuration and facilitates configuration by automatically associating the filtered SRD, and other configuration elements associated with the filtered SRD, wherever applicable.

Multiple SRDs for Multi-tenant Deployments

The device can be deployed in a multi-tenant architecture, serving multiple customers (tenants) from a single, shared physical entity. The device's multi-tenant feature is fully scalable, offering almost “non-bleeding” partition per tenant, whereby users of one tenant can’t infringe on the space of users of another tenant. The device provides per tenant configuration, monitoring,

reporting, analytics, alarms and interfacing. The device is a real-time multi-tenant system that provides each tenant with optimal real-time performance, as each session received by the device is classified and processed only through the tenant's "orbit".

While some enterprises are large enough to justify a dedicated standalone device, many enterprises require only a fraction of the device's capacity and capabilities. Service providers offering SIP Trunking services can funnel multiple enterprises into a single device and thereby, reap significant cost improvements over a device-per-customer model. Tenant size in a multi-tenant architecture can vary and therefore, the instance CPU, memory and interface allocations should be optimized so as not to waste resources for small-sized tenants on the one hand, and not to allocate too many instances for a single tenant/customer on the other. For example, it would be a waste to allocate a capacity of 100 concurrent sessions to a small tenant for which 10 concurrent sessions suffice.

In a multi-tenant deployment, each tenant is represented by a dedicated SRD. The different Layer-3 networks (e.g., LAN IP-PBX users, WAN SIP Trunk, and WAN far-end users) of the tenant are represented by SIP Interfaces, which are all associated with the tenant's SRD. As related configuration entities (SIP Interfaces, IP Groups, Proxy Sets, Classification rules, and IP-to-IP Routing rules) are associated with the specific SRD, each SRD has its own logically separated configuration tables (although configured in the same tables). Therefore, full logical separation (on the SIP application layer) between tenants is achieved by SRD.

To create a multi-tenant configuration topology that is as non-bleeding as possible, you can configure an SRD (tenant) as *Isolated* and *Shared*:

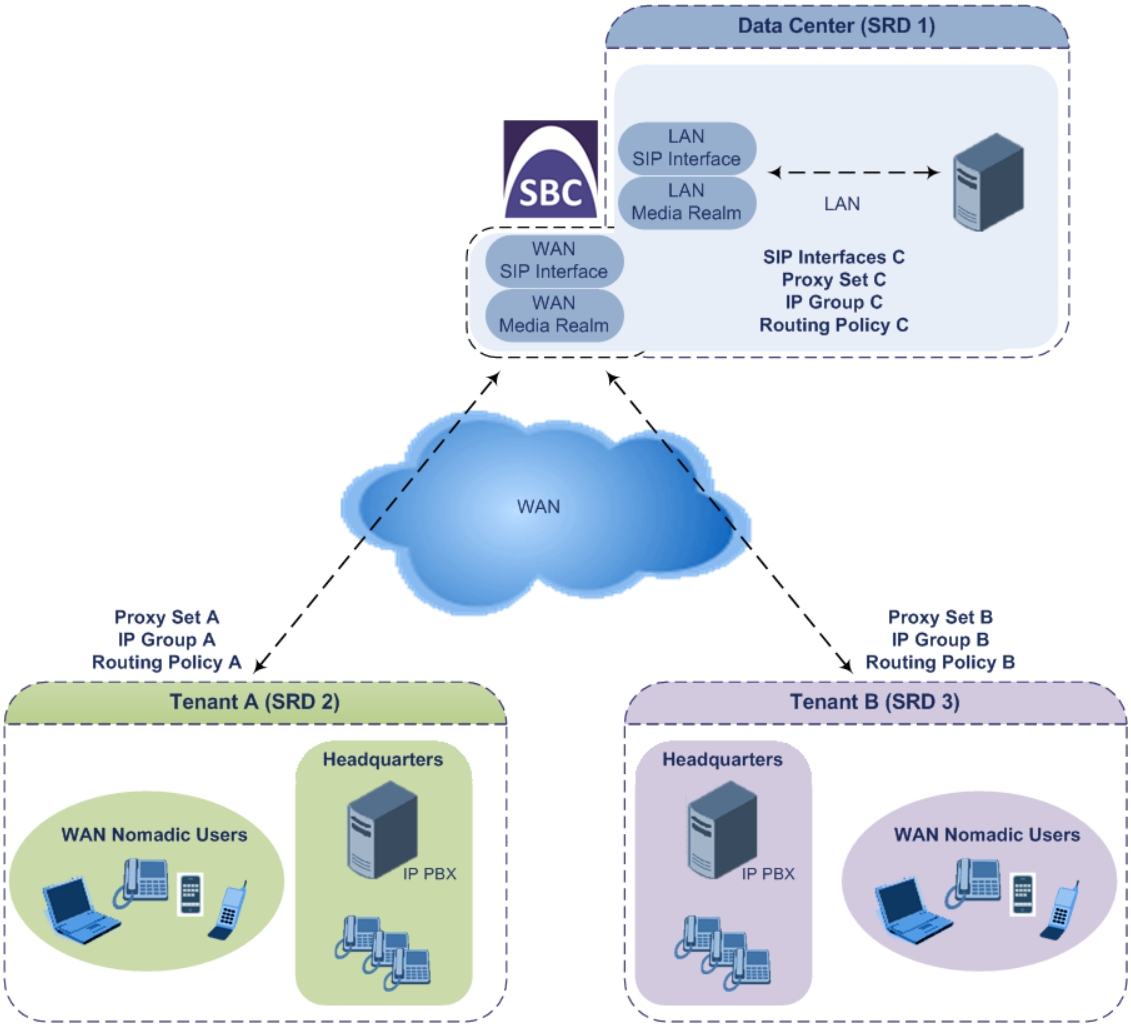
- **Isolated SRD:** An Isolated SRD has its own dedicated SIP resources (SIP Interfaces, Proxy Sets, and IP Groups). No other SRD can use the SIP resources of an Isolated SRD. Thus, call traffic of an Isolated SRD is kept separate from other SRDs (tenants), preventing any risk of traffic "leakage" with other SRDs.

  Isolated SRDs are more relevant when each tenant needs its own separate (dedicated) routing "table" for non-bleeding topology. Separate routing tables are implemented using Routing Policies. In such a non-bleeding topology, routing between Isolated SRDs is not possible. This enables accurate and precise routing per SRD, eliminating any possibility of erroneous call routing between SRDs, restricting routing to each tenant's (SRD's) sphere. Configuring only one Routing Policy that is shared between Isolated SRDs is not best practice for non-bleeding environments, since it allows routing between these SRDs.

- **Shared SRD:** Isolated SRDs have their own dedicated SIP resources (SIP Interfaces, Proxy Sets, and IP Groups). This may not be possible in some deployments. For example, in deployments where all tenants use the same SIP Trunking service, or use the same SIP Interface due to limited SIP interface resources (e.g., multiple IP addresses cannot be allocated and SIP port 5060 must be used). In contrast to Isolated SRDs, a Shared SRD can share its' SIP resources with all other SRDs (Shared and Isolated). This is typically required when tenants need to use common resources. In the SIP Trunk example, the SIP Trunk would be associated with a Shared SRD, enabling all tenants to route calls with the SIP Trunk.

Another configuration entity that can be used for multi-tenant deployments is the Routing Policy. Routing Policies allow each SRD (or tenant) to have its own routing rules, manipulation rules, Least Cost Routing (LCR) rules, and/or LDAP-based routing configuration. However, not all multi-tenant deployments need multiple Routing Policies and typically, their configuration is not required. Isolated SRDs are more relevant only when each tenant requires its own dedicated Routing Policy to create separate, dedicated routing "tables"; for all other scenarios, SRDs can be Shared. For more information on Routing Policies, see Configuring SBC Routing Policy Rules.

The figure below illustrates a multi-tenant architecture with Isolated SRD tenants ("A" and "B") and a Shared SRD tenant ("Data Center") serving as a SIP Trunk:



To facilitate multi-tenant configuration through CLI, you can access a specific tenant "view". Once in a specific tenant view, all configuration commands apply only to the currently viewed tenant. Only table rows (indexes) belonging to the viewed tenant can be modified. New table rows are automatically associated with the viewed tenant (i.e., SRD name). The display of tables and show running-configuration commands display only rows relevant to the viewed tenant (and shared tenants). The show commands display only information relevant to the viewed tenant. To support this CLI functionality, use the following commands:

- To access a specific tenant view:

```
# srd-view <SRD name>
```

Once accessed, the tenant's name (i.e., SRD name) forms part of the CLI prompt, for example:

```
# srd-view datacenter
(srd-datacenter)#
```

- To exit the tenant view:

```
# no srd-view
```

Cloning SRDs

You can clone (duplicate) existing SRDs. This is especially useful when operating in a multi-tenant environment and you need to add new tenants (SRDs). The new tenants can quickly and easily be added by simply cloning one of the existing SRDs. Once cloned, all you need to do is tweak configuration entities associated with the SRD clone.

When an SRD is cloned, the device adds the new SRD clone to the next available index row in the SRDs table. The SRD clone is assigned a unique name in the following syntax format: <unique clone ID>_<original SRD index>_CopyOf_<name, or index if no name, of original SRD>. For example, if you clone SRD "SIP-Trunk" at index 2, the new SRD clone is assigned the name, "36454371_2_CopyOf_SIP-Trunk".

The SRD clone has identical settings as the original SRD. In addition, all configuration entities associated with the original SRD are also cloned and these clones are associated with the SRD clone. The naming convention of these entities is the same as the SRD clone (see above) and all have the same unique clone ID ("36454371" in the example above) as the cloned SRD. These configuration entities include IP Groups, SIP Interfaces, Proxy Sets (without addresses), Classification rules, and Call Admission Control profiles. If the Routing Policy associated with the original SRD is not associated with any other SRD, the Routing Policy is also cloned and its' clone is associated with the SRD clone. All configuration entities associated with the original Routing Policy are also cloned and these clones are associated with the Routing Policy clone. These configuration entities include IP-to-IP Routing rules, Inbound Manipulation rules, and Outbound Manipulation rules.

When any configuration entity is cloned (e.g., an IP-to-IP Routing rule) as a result of a cloned SRD, all fields of the entity's row which "point" to other entities (e.g., SIP Interface, Source IP Group, and Destination IP Group) are replaced by their corresponding clones.

For some cloned entities such as SIP Interfaces, some parameter values may change. This occurs in order to avoid the same parameter having the same value in more than one table row (index), which would result in invalid configuration. For example, a SIP Interface clone will have an empty Network Interface setting. After the clone process finishes, you thus need to update the Network Interface for valid configuration.

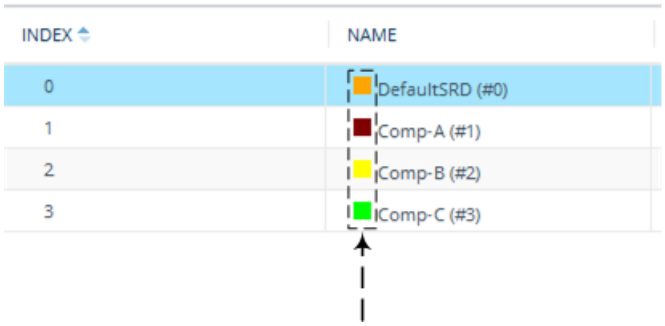
➢ **To clone an SRD:**

- Web interface: In the SRDs table, select an SRD to clone, and then click the **Clone** button.
- CLI:

```
(config-voip)# srd clone <SRD index that you want cloned>
```

Color-Coding of SRDs in Web Interface

To easily identify your configured SRDs, the Web interface displays each SRD in a unique color. The color is automatically and randomly assigned to new SRDs and is displayed in a box alongside the name of the SRD in tables where the SRD is configured or assigned. This is applied throughout the Web interface. The following example shows SRDs assigned with unique color codes.



Automatic Configuration based on SRD

To facilitate configuration and eliminate possible flaws in configuration due to invalid associations between configuration entities, the Web interface automatically configures configuration entities based on SRD:

- If you delete an SRD (in the SRDs table) that is associated with other configuration entities in other tables, the device automatically deletes the associated table rows. For example, if you delete an SRD that is associated with a Proxy Set, the device automatically deletes the Proxy Set.
- If you associate an SRD with a configuration entity in another table (i.e., other than the SRDs table), the device automatically configures certain parameters of the configuration entity according to the SRD or associated SRD. For example, if you add a rule in the IP-to-IP

Routing table and you select a Routing Policy, the 'Source IP Group' and 'Destination IP Group' parameters list only IP Groups that re associated with the SRD to which the Routing Policy is assigned (and IP Groups belonging to a Shared SRD, if exists).

- If your configuration setup includes only a single SRD, the device automatically selects the SRD when adding related configuration entities. For example, when adding an IP Group, the single SRD is automatically selected in the Add Row dialog box.

Configuring SIP Interfaces

The SIP Interfaces table lets you configure up to 500 SIP Interfaces. A SIP Interface represents a Layer-3 network in your deployment environment, by defining a local, listening port number and type (e.g., UDP), and assigning an IP network interface for SIP signaling traffic. For example, if your deployment consists of an IP PBX in the LAN, a SIP Trunk in the WAN, and remote far-end users in the WAN, you would need to configure a SIP Interface for each of these SIP entities. You can also configure various optional features for the SIP Interface such as assigning it a Media Realm, blocking calls received on the SIP Interface from users not registered with the device, and enabling direct media (media bypass).

Each SIP Interface can be associated with only one SRD. As the SRD configuration entity represents your VoIP deployment SIP network (Layer 5), you need to associate your SIP Interfaces with a specific SRD in order to represent your Layer-3 networks. For most deployments (except multi-tenant deployments), your SRD represents your entire network and thus, only one SRD is required. The device provides a default SRD and in such scenarios where only a single SRD is required, your SIP Interfaces are automatically assigned to the default SRD. Therefore, there is no need to even handle SRD configuration entity.

Once configured, you can apply SIP Interfaces to calls, by assigning them to the following configuration entities in their respective tables:

- (Mandatory) Proxy Set to specify the SIP Interface for communication with the proxy server (i.e., IP Group). For more information, see Configuring Proxy Sets.
- Intrusion Detection System (IDS) for applying the IDS policy to a specific SIP Interface. For more information, see Configuring IDS Policies.
- IP-to-IP Routing rules for specifying the destination SIP Interface to where you want to route the call. For more information, see Configuring SBC IP-to-IP Routing Rules.
- Classification rules for specifying the SIP Interface as a matching characteristic of the incoming call. This is especially useful for the single SRD-configuration topology, where each SIP Interface represents a Layer-3 network (SIP entity). Therefore, classification of calls to IP Groups (SIP entities) can be based on SIP Interface.

  The SIP Interface can also be used for tag-based classification of incoming SIP dialogs if the SIP Interface is configured with a Call Setup Rule Set ID that determines the source tag. For more information, see Configuring Classification Based on Tags on page 714.

  For more information on classification, see Configuring Classification Rules.

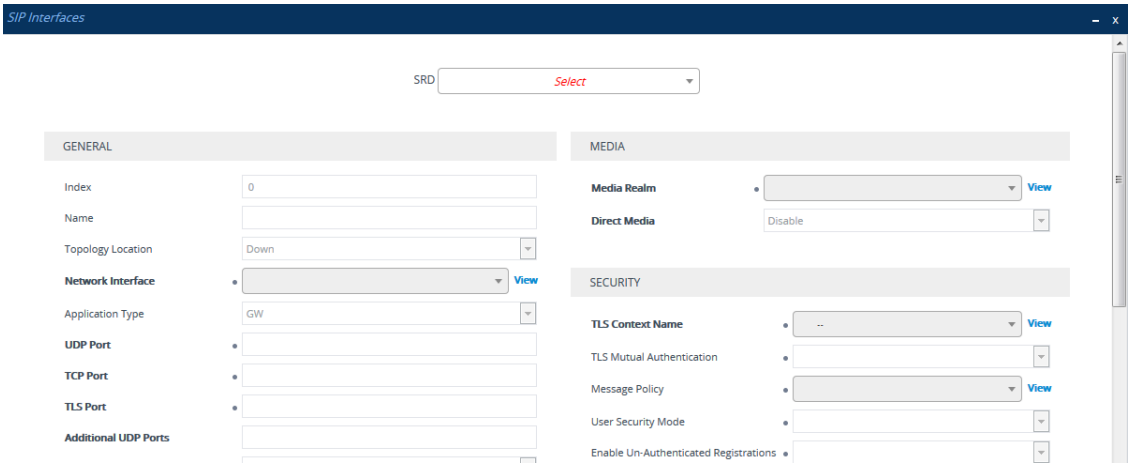
- Tel-to-IP Routing rules for specifying the destination SIP Interface to where you want to route Tel-to-IP calls. For more information, see Configuring Tel-to-IP Routing Rules.
- IP-to-Trunk Group Routing rules for specifying the SIP Interface as a matching characteristics for the incoming IP call.

> The device terminates active calls associated with a SIP Interface if you do one of the following:
> - Delete the associated SIP Interface.
> - Edit any of the following fields of the associated SIP Interface: 'Application Type', 'UDP Port, 'TCP Port', 'TLS Port' or 'SRD' fields.
> - Edit or delete a network interface in the IP Interfaces table that is associated with the SIP Interface.

The following procedure describes how to configure SIP interfaces through the Web interface. You can also configure it through ini file [SIPInterface] or CLI (`configure voip > sip-interface`).

➢ To configure a SIP Interface:

1. Open the SIP Interfaces table (**Setup** menu > **Signaling & Media** tab > **Core Entities** folder > **SIP Interfaces**).
2. Click **New**; the following dialog box appears:



3. Configure a SIP Interface according to the parameters described in the table below.
4. Click **Apply**.

Table 18-5: SIP Interfaces table Parameter Descriptions

Parameter	Description
'SRD' `srd-name` [SIPInterface_SRDName]	Assigns an SRD to the SIP Interface. If only one SRD is configured in the SRDs table, the SRD is assigned to the SIP Interface by

Parameter	Description
	default. If multiple SRDs are configured in the SRDs table, no value is defined and you must assign an SRD. To configure SRDs, see Configuring SRDs. **Note:** ■ The parameter is mandatory. ■ You can assign the same SRD to multiple SIP Interfaces .
General	
'Index' [SIPInterface_Index]	Defines an index for the new table row. **Note:** Each row must be configured with a unique index.
'Name' `interface-name` [SIPInterface_InterfaceName]	Defines a descriptive name, which is used when associating the row in other tables. The valid value is a string of up to 40 characters. By default, if you do not configure a name, the device automatically assigns the name "SIPInterface_<row index>" (e.g., "SIPInterface_1" when added to Index 1). **Note:** The parameter value cannot contain a forward slash (/).
'Topology Location' `topology-location` [SIPInterface_TopologyLocation]	Defines the display location of the SIP Interface in the Topology view in the Web interface. ■ [0] **Down** = (Default) The SIP Interface element is displayed on the lower border of the view. ■ [1] **Up** = The SIP Interface element is displayed on the upper border of the view. For more information on the Topology view, see Building and Viewing SIP Entities in Topology View.
'Network Interface' `network-interface` [SIPInterface_NetworkInterface]	Assigns an IP Interface to the SIP Interface. By default, no value is defined. To configure IP Interfaces, see Configuring IP Network Interfaces.

Parameter	Description
	Note: ■ The parameter is mandatory. ■ The 'Application Type' parameter of the assigned IP Interface must at least include "Control".
'Application Type' `application-type` [SIPInterface_ApplicationType]	Defines the application for which the SIP Interface is used. ■ [2] **SBC** = SBC application.
'UDP Port' `udp-port` [SIPInterface_UDPPort]	Defines the device's listening and source port for SIP signaling traffic over UDP. The valid range is 1 to 65534. The default is 5060. **Note:** ■ The port number **must** be different from ports configured for RTP traffic (i.e., ports configured for Media Realms and Media Realm Extensions) using the same IP network interface. For example, if the RTP port range is 6000 to 6999, the SIP port can either be less than 6000 or greater than 6999. ■ Each SIP Interface must have a unique UDP signaling port within its underlying network interface (i.e., no port overlapping between such SIP Interfaces). For example: ✓ Valid configuration: • SIP Interface #0: 'UDP Port' = 6010; 'Network Interface' = #0 • SIP Interface #1: 'UDP Port' = 6010; 'Network Interface' = #1 ✓ Invalid configuration: • SIP Interface #0: 'UDP Port' = 6010; 'Network Interface' = #0 • SIP Interface #1: 'UDP Port' = 6010; 'Network Interface' = #0

Parameter	Description
'TCP Port' `tcp-port` [SIPInterface_TCPPort]	Defines the device's listening port for SIP signaling traffic over TCP. The valid range is 1 to 65534. The default is 5060. **Note:** ■ For the specific SIP Interface, the TCP port number must be different from the TLS port number (configured by the 'TLS Port' parameter below). ■ The port must be different from the TCP port configured for Media Realms and Media Realm Extensions that use the same IP Interface. ■ The source ports used for outgoing TCP connections are not configurable and are dynamically determined by the device in the range of 32,768-61,000. ■ Each SIP Interface must have a unique TCP signaling port within its underlying network interface (i.e., no port overlapping between such SIP Interfaces). For example: ✓ Valid configuration: • SIP Interface #0: 'TCP Port' = 6010; 'Network Interface' = #0 • SIP Interface #1: 'TCP Port' = 6010; 'Network Interface' = #1 ✓ Invalid configuration: • SIP Interface #0: 'TCP Port' = 6010; 'Network Interface' = #0 • SIP Interface #1: 'TCP Port' = 6010; 'Network Interface' = #0
'TLS Port' `tls-port` [SIPInterface_TLSPort]	Defines the device's listening port for SIP signaling traffic over TLS. The valid range is 1 to 65534. The default is 5061. **Note:**

Parameter	Description
	■ For the specific SIP Interface, the TLS port number must be different from the TCP port number (configured by the 'TCP Port' parameter above). ■ The port must be different from the TCP port configured for Media Realms and Media Realm Extensions that use the same IP Interface. ■ The source ports used for outgoing TLS connections are not configurable and are dynamically determined by the device in the range of 32,768-61,000. ■ Each SIP Interface must have a unique TLS signaling port within its underlying network interface (i.e., no port overlapping between such SIP Interfaces). For example: ✓ Valid configuration: • SIP Interface #0: 'TLS Port' = 6020; 'Network Interface' = #0 • SIP Interface #1: 'TLS Port' = 6020; 'Network Interface' = #1 ✓ Invalid configuration: • SIP Interface #0: 'TLS Port' = 6020; 'Network Interface' = #0 • SIP Interface #1: 'TLS Port' = 6020; 'Network Interface' = #0
'Additional UDP Ports' `additional-udp-ports` [SIPInterface_AdditionalUDPPorts]	Defines a port range for the device's local, listening and source ports for SIP signaling traffic over UDP. The parameter can be used for the following features: ■ Assigning a unique port per registered user (User-type IP Group) on the leg interfacing with the proxy server (Server-type IP Group). For enabling this feature and for more information, see the 'User UDP Port Assignment' parameter in the IP Groups table.

Parameter	Description
	■ Assigning a specific local port to each SIP entity (e.g., PBX) communicating with a common SIP entity (e.g., proxy server). This is the port on the leg interfacing with the proxy server. In other words, the SIP Interface associated with the proxy server. For more information, see Configuring Specific UDP Ports using Tag-based Routing. ■ Assigning a unique port for each Account registering with the same Serving IP Group (registrar server). For more information, see Configuring Registration Accounts on page 573. The valid range is 1,025 to 65535. The range is configured using the syntax *x*-*y*, where *x* is the starting port and *y* the ending port of the range (e.g., 6000-7000). By default, the parameter is not configured. **Note:** ■ To configure whether the device keeps the configured ports (sockets) open or opens them only when needed, use the SIP Interface's 'Additional UDP Ports Mode' parameter (below). ■ The parameter's port range value must not overlap with the UDP port configured by the 'UDP Port' parameter (SIPInterface_ UDPPort). For example, if the 'UDP Port' parameter is configured to 5070, you cannot configure the 'Additional UDP Ports' parameter with a range of 5060-6000. ■ The parameter's port range value must not overlap with UDP port ranges of Media Realms and Media Realm Extensions that are configured on the same network interface. For example, if the RTP port range is 6000-6999, you must configure the 'Additional UDP Ports' parameter to a range that is less than 6000 or greater than 6999.

Parameter	Description
	■ The maximum number of ports in the range is limited to the maximum number of licensed registered SBC users as specified in the License Key installed on the device, or the maximum number of IP Groups that can be configured (see Configuring IP Groups) - the higher of the two determines it. For example, if the License Key allows 20 users and the maximum IP Groups that can be configured is 10, then the maximum number of ports is 20.
'Additional UDP Ports Mode' `additional-udp-ports-mode` [AdditionalUDPPortsMode]	Enables the device to open sockets (ports) for signaling only when needed. The parameter applies to the Additional UDP Port feature with dynamic port allocation (see the 'Additional UDP Ports' parameter, above). This allows you to configure the additional UDP port range without having to make sure that the total number of configured ports are within the maximum, as defined by the device's License Key. ■ [0] **Always Open** = (Default) The device keeps the ports (sockets) that are configured in the SIP Interface's 'Additional UDP Ports' parameter, open all the time. ■ [1] **Open When Used** = For the ports (sockets) that are configured in the SIP Interface's 'Additional UDP Ports' parameter, the device opens a port only when it is used. A port is needed when the device initiates registration with an external SIP entity for a SIP Account (sent to the Account's Serving IP Group), or forwards a registration request from a user (IP Group) to a proxy (Server-type IP Group). This option is applicable only to dynamic port allocation, where a port is allocated on the outgoing REGISTER message and closed when the registration expires. For HA systems, upon a switchover, all the ports used in the active device are also opened on

Parameter	Description
	the redundant device (now active), so that the SIP entity is reachable. Ports that are not configured by the SIP Interface's 'Additional UDP Ports' parameter are closed. The option is applicable only when the SIP Interface's 'Additional UDP Ports' parameter is configured and enabled for a Server-type IP Group (IP Group's 'User UDP Port Assignment' parameter) and/or SIP Account (Account's 'UDP Port Assignment' parameter). **Note:** ■ For static port allocation (i.e., using additional UDP ports feature for assigning a specific local port to each SIP entity), configure the parameter to **Always Open**.
'Encapsulating Protocol' `encapsulating-protocol` [SIPInterface_EncapsulatingProtocol]	Defines the type of incoming traffic (SIP messages) expected on the SIP Interface. ■ [0] **No Encapsulation** = (Default) Regular (non-WebSocket) traffic. ■ [1] **WebSocket** = Traffic received on the SIP Interface is identified by the device as WebSocket signaling traffic (encapsulated by WebSocket frames). For outgoing traffic, the device encapsulates the traffic using the WebSocket protocol (frames) on the TCP/TLS ports. For more information on WebSocket, see SIP over WebSocket. **Note:** WebSocket encapsulation is not supported for UDP ports.
'Enable TCP Keepalive' `tcp-keepalive-enable` [SIPInterface_TCPKeepaliveEnable]	Enables the TCP Keep-Alive mechanism with the IP entity on this SIP Interface. TCP keep-alive can be used, for example, to keep a NAT entry open for clients located behind a NAT server, or simply to check that the connection to the IP entity is available. ■ [0] **Disable** (default)

Parameter	Description
	■ [1] **Enable** **Note:** To configure TCP keepalive, use the following ini file parameters: TCPKeepAliveTime, TCPKeepAliveInterval, and TCPKeepAliveRetry.
'Used By Routing Server' `used-by-routing-server` [SIPInterface_UsedByRoutingServer]	Enables the SIP Interface to be used by a third-party routing server for call routing decisions. ■ [0] **Not Used** (default) ■ [1] **Used** For more information on the third-party routing server feature, see Centralized Third-Party Routing Server.
'Pre-Parsing Manipulation Set' `pre-parsing-man-set` [SIPInterface_PreParsingManSetName]	Assigns a Pre-Parsing Manipulation Set to the SIP Interface. This lets you apply pre-parsing SIP message manipulation rules on any incoming SIP message received on this SIP Interface. By default, no Pre-Parsing Manipulation Set is assigned. To configure Pre-Parsing Manipulation Sets, see Configuring Pre-parsing Manipulation Rules. **Note:** ■ Pre-Parsing Manipulation is done only on incoming calls. ■ The device performs Pre-Parsing Manipulation before Pre-Classification Manipulation and Classification.
'CAC Profile' `cac-profile` [SIPInterface_AdmissionProfile]	Assigns a Call Admission Control Profile (CAC rules) to the SIP Interface. By default, no value is defined. To configure CAC Profiles, see Configuring Call Admission Control on page 696.
Classification	
'Classification Failure Response Type' `classification_fail_response_type`	Defines the SIP response code that the device sends if a received SIP request (OPTIONS, REGISTER, or INVITE) fails the SBC Classification

Parameter	Description
[SIPInterface_ ClassificationFailureResponseType]	process. The valid value can be a SIP response code from 400 through 699, or it can be set to 0 to not send any response at all. The default response code is 500 (Server Internal Error). This feature is important for preventing Denial of Service (DoS) attacks, typically initiated from the WAN. Malicious attackers can use SIP scanners to detect ports used by SIP devices. These scanners scan devices by sending UDP packets containing a SIP request to a range of specified IP addresses, listing those that return a valid SIP response. Once the scanner finds a device that supports SIP, it extracts information from the response and identifies the type of device (IP address and name) and can execute DoS attacks. A way to defend the device against such attacks is to not send a SIP reject response to these unclassified "calls" so that the attacker assumes that no device exists at such an IP address and port. **Note:** ■ The parameter is applicable only if you configure the device to reject unclassified calls, which is done using the 'Unclassified Calls' parameter (see Configuring Classification Rules).
'Pre Classification Manipulation Set ID' `preclassification-manset` [SIPInterface_ PreClassificationManipulationSet]	Assigns a Message Manipulation Set ID to the SIP Interface. This lets you apply SIP message manipulation rules on incoming SIP initiating-dialog request messages (not in-dialog), received on this SIP Interface, prior to the Classification process. By default, no Message Manipulation Set ID is defined. To configure Message Manipulation rules, see Configuring SIP Message Manipulation. **Note:** ■ The Message Manipulation Set assigned to a SIP Interface that is associated with an

Parameter	Description
	outgoing call, is ignored. Only the Message Manipulation Set assigned to the associated IP Group is applied to the outgoing call. ■ If both the SIP Interface and IP Group associated with the incoming call are assigned a Message Manipulation Set, the one assigned to the SIP Interface is applied first. ■ If Classification fails or the request is rejected prior to the Classification stage, then manipulation rules according to this parameter are applied to the reject response. In this case, the device adds a Reason header to the reject response. If routing fails, manipulation on the reject response is according to the 'Outbound Message Manipulation Set' parameter of the classified IP Group. When a Reason header is added to the reject response, its value is according to the type of failure: ✓ Routing failure: "General Routing Failure" ✓ Classification failure: "Classification Failure" ✓ Pre-Classification rejection due to device overload: "Board In Overload" ✓ Pre-Classification rejection due to locked device: "Board Is Locked" ✓ Pre-Classification rejection due to too many SIP headers in the request: "Header Overflow" ✓ Post-Classification failure of a REGISTER request when the source IP Group doesn't allow registers from the IP Group: "IPGroup Registration Mode Configuration"
'Call Setup Rules Set ID' `call-setup-rules-set-id`	Assigns a Call Setup Rules Set ID to the SIP Interface. The Call Setup Rule is run before the

Parameter	Description
[SIPInterface_CallSetupRulesSetId]	Classification stage. By default, no Call Setup Rules Set ID is defined. To configure Call Setup Rules, see Configuring Call Setup Rules on page 595. Call Setup Rules can be used for Classification of incoming calls to IP Groups based on tags (source), as described in Configuring Classification Based on Tags on page 714. **Note:** ■ Call Setup Rules that are triggered from the SIP Interfaces table are done after identifying the incoming SIP Interface, but before classification, manipulation and routing. It can run synchronous operations including Dial Plan queries, but it can't run asynchronous queries (LDAP, ENUM, and HTTP). ■ Call Setup Rules can be used to generated source and destination tags. For classification, only source tags are used. ■ Using Call Setup Rules with the SIP Interface is suitable for actions that affect the source and the Classification of SIP dialog requests (such as modifying source tags or modifying the From header). It's not suitable for actions that affect the destination of the request and its routing (such as modifying the Request-URI header) because it might conflict with other features.
Media	
'Media Realm' `media-realm-name` [SIPInterface_MediaRealm]	Assigns a Media Realm to the SIP Interface. By default, no value is defined. To configure Media Realms, see Configuring Media Realms.
'Direct Media' `sbc-direct-media` [SIPInterface_SBCDirectMedia]	Enables direct media (RTP/SRTP) flow or media bypass (i.e., no Media Anchoring) between endpoints associated with the SIP Interface for SBC calls.

Parameter	Description
	■ [0] **Disable** = (Default) Media Anchoring is employed, whereby the media stream traverses the device (and each leg uses a different coder or coder parameters). ■ [1] **Enable** = Direct Media is enabled (i.e., no Media Anchoring). Media stream flows directly between the endpoints (i.e., doesn't traverse the device). ■ [2] **Enable when Same NAT** = Direct Media is enabled (i.e., no Media Anchoring). Media stream flows directly between the endpoints if they are located behind the same NAT. For more information on direct media, see Direct Media. **Note:** ■ If the parameter is enabled for direct media and the two endpoints belong to the same SIP Interface, calls cannot be established if the following scenario exists: ✓ One of the endpoints is defined as a foreign user (for example, "follow me service") ✓ and one endpoint is located on the WAN and the other on the LAN. The reason for the above is that in direct media, the device does not interfere in the SIP signaling such as manipulation of IP addresses, which is necessary for calls between LAN and WAN. ■ To enable direct media for all calls, use the global parameter [SBCDirectMedia]. If enabled, even if the SIP Interface is disabled for direct media, direct media is employed for calls belonging to the SIP Interface. ■ If you enable direct media for the SIP Interface, make sure that your Media Realm provides sufficient ports, as media may traverse the device for mid-call services

Parameter	Description
	(e.g., call transfer). ■ If you have configured a SIP Recording rule (see SIP-based Media Recording on page 247) for calls associated with this SIP Interface, the device automatically disables direct media for these calls (during their SIP signaling setup). This ensures that the media passes through the device so that it can be recorded and sent to the SRS. However, if you enable direct media using the [SBCDirectMedia] global parameter (i.e., for all calls), direct media is always enforced and calls will not be recorded.
Security	
'TLS Context Name' `tls-context-name` [SIPInterface_TLSContext]	Assigns a TLS Context (TLS configuration) to the SIP Interface. The default TLS Context ("default" at Index 0) is assigned to the SIP Interface by default. **Note:** ■ For incoming calls: The assigned TLS Context is used if no TLS Context is configured for the Proxy Set associated with the call or classification to an IP Group based on Proxy Set fails. ■ For outgoing calls: The assigned TLS Context is used if no TLS Context is configured for the Proxy Set associated with the call. ■ To configure TLS Contexts, see Configuring TLS Certificates on page 158.
'TLS Mutual Authentication' `tls-mutual-auth` [SIPInterface_TLSMutualAuthentication]	Enables TLS mutual authentication for the SIP Interface (when the device acts as a server). ■ [0] **Disable** = Device does not request the client certificate for TLS connection on the SIP Interface. ■ [1] **Enable** = Device requires receipt and verification of the client certificate to establish the TLS connection on the SIP

Parameter	Description
	Interface. By default, no value is defined and the [SIPSRequireClientCertificate] global parameter setting is applied.
'Message Policy' `message-policy-name` [SIPInterface_MessagePolicyName]	Assigns a SIP message policy to the SIP interface. To configure SIP Message Policy rules, see Configuring SIP Message Policy Rules.
'User Security Mode' `block-un-reg-users` [SIPInterface_BlockUnRegUsers]	Defines the blocking (reject) policy for incoming SIP dialog-initiating requests (e.g., INVITE messages) from registered and unregistered users belonging to the SIP Interface. ■ [-1] **Not Configured** = (Default) The corresponding parameter in the SRDs table (SRD_BlockUnRegUsers) of the SRD that is associated with the SIP Interface is applied. ■ [0] **Accept All** = Accepts requests from registered and unregistered users. ■ [1] **Accept Registered Users** = Accepts requests only from users registered with the device. Requests from users not registered are rejected. ■ [2] **Accept Registered Users from Same Source** = Accepts requests only from registered users whose source address is the same as that registered with the device (during the REGISTER message process). All other requests are rejected. If the transport protocol is UDP, the device verifies the IP address and port; otherwise, it verifies only the IP address. The verification is performed before any of the device's call handling processes (i.e., Classification, Manipulation and Routing). **Note:** ■ The parameter is applicable only to calls belonging to User-type IP Groups.

Parameter	Description
	■ The feature is not applicable to REGISTER requests. ■ The option, Accept Registered Users from Same Source [2] does not apply to registration refreshes. These requests are accepted even if the source address is different to that registered with the device. ■ When the device rejects a call, it sends a SIP 500 "Server Internal Error" response to the user. In addition, it reports the rejection (Dialog establish failure - Classification failure) using the Intrusion Detection System (IDS) feature (see Configuring IDS Policies), by sending an SNMP trap. ■ If you configure the parameter to any value other than default [-1], it overrides the corresponding parameter in the SRDs table (SRD_BlockUnRegUsers) for the SRD associated with the SIP Interface.
'Enable Un-Authenticated Registrations' `enable-un-auth-registrs` [SIPInterface_ EnableUnAuthenticatedRegistrations]	Enables the device to accept REGISTER requests and register them in its registration database from new users that have not been authenticated by a proxy/registrar server (due to proxy down) and thus, re-routed to a User-type IP Group. In normal operation scenarios in which the proxy server is available, the device forwards the REGISTER request to the proxy and if authenticated by the proxy (i.e., device receives a success response), the device adds the user to its registration database. The routing to the proxy is according to the SBC IP-to-IP Routing table where the destination is the proxy's IP Group. However, when the proxy is unavailable (e.g., due to network connectivity loss), the device can accept REGISTER requests from new users if a matching alternative routing rule exists in the SBC IP-to-IP Routing table where the destination is the user's User-type IP Group (i.e., call survivability scenarios) and if the parameter is enabled.

Parameter	Description
	■ [-1] **Not Configured** = (Default) The corresponding parameter in the SRDs table (SRD_EnableUnAuthenticatedRegistrations) of the SRD associated with the SIP Interface is applied. ■ [0] **Disable** = The device rejects REGISTER requests from new users that were not authenticated by a proxy server. ■ [1] **Enable** = The device accepts REGISTER requests from new users even if they were not authenticated by a proxy server, and registers the user in its registration database. **Note:** ■ Regardless of the parameter, the device always accepts registration refreshes from users that are already registered in its database. ■ If configured to Disable or Enable, the parameter overrides the 'Enable Un-Authenticated Registrations' parameter settings of the SRD (in the SRDs table) that is associated with the SIP Interface.
'Max. Number of Registered Users' `max-reg-users` [SIPInterface_MaxNumOfRegUsers]	Defines the maximum number of users belonging to the SIP Interface that can register with the device. By default, no value is defined (i.e., the number of allowed user registrations is unlimited).

Configuring IP Groups

The IP Groups table lets you configure up to 700 IP Groups. An IP Group represents a SIP entity in the network with which the device communicates. This can be a server (e.g., IP PBX or ITSP) or a group of users (e.g., LAN IP phones). For servers, the address of the IP Group is typically defined by associating it with a Proxy Set (see Configuring Proxy Sets).

You can use IP Groups for the following:

- Classification of incoming SIP dialog-initiating requests (e.g., INVITE messages) to IP Groups based on Proxy Set. If the source address of the incoming SIP dialog is defined for a Proxy

Set, the assigns ("bonds") the SIP dialog to the IP Group associated with the Proxy Set. The feature is configured using the IP Groups table's 'Classify by Proxy Set' parameter. For more information and recommended security guidelines, see the parameter's description, later in this section.

- Classification of incoming SIP dialog-initiating requests (e.g., INVITE messages) to IP Groups based on source tags of incoming dialog. Tag-based classification occurs only if Classification based on user registration and on Proxy Sets fail. For more information, see Configuring Classification Based on Tags on page 714.
- Representing the source and destination of the call in IP-to-IP Routing rules (see Configuring SBC IP-to-IP Routing Rules).
- SIP dialog registration and authentication (digest user/password) of specific IP Groups (Served IP Group, e.g., corporate IP-PBX) with other IP Groups (Serving IP Group, e.g., ITSP). This is configured in the Accounts table (see Configuring Registration Accounts).
- Included in routing decisions by a third-party routing server. If deemed necessary for routing, the routing server can even create an IP Group. For more information, see Centralized Third-Party Routing Server.

You can also apply the device's Quality of Experience feature to IP Groups:

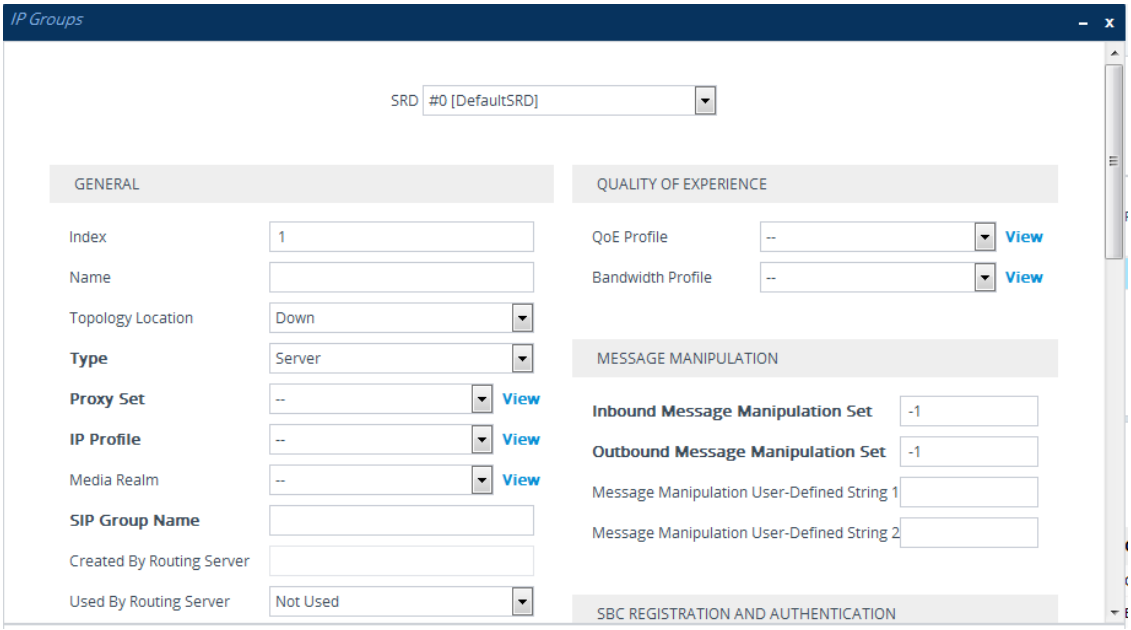
- **Quality of Experience Profile:** Call quality monitoring based on thresholds for voice metrics (e.g., MOS) can be applied per IP Group. For example, if MOS is considered poor, calls belonging to this IP Group can be rejected. To configure Quality of Experience Profiles, see Configuring Quality of Experience Profiles.
- **Bandwidth Profile:** Bandwidth utilization thresholds can be applied per IP Group. For example, if bandwidth thresholds are crossed, the device can reject any new calls on this IP Group. To configure Bandwidth Profiles, see Configuring Bandwidth Profiles.

- If you delete an IP Group or modify the 'Type' or 'SRD' parameters, the device immediately terminates currently active calls that are associated with the IP Group. In addition, all users belonging to the IP Group are removed from the device's users database.

The following procedure describes how to configure IP Groups through the Web interface. You can also configure it through ini file [IPGroup] or CLI (`configure voip > ip-group`).

➢ **To configure an IP Group:**

1. Open the IP Groups table (**Setup** menu > **Signaling & Media** tab > **Core Entities** folder > **IP Groups**).
2. Click **New**; the following dialog box appears:



3. Configure an IP Group according to to the parameters described in the table below.
4. Click **Apply**.

Table 18-6: IP Groups Table Parameter Descriptions

Parameter	Description
'SRD' srd-name [IPGroup_SRDName]	Assigns an SRD to the IP Group. If only one SRD is configured in the SRDs table, the SRD is assigned by default. If multiple SRDs are configured in the SRDs table, no value is assigned by default and you must assign one. To configure SRDs, see Configuring SRDs. **Note:** The parameter is mandatory.
General	
'Index' [IPGroup_Index]	Defines an index for the new table row. **Note:** ■ Each row must be configured with a unique index.
'Name' name [IPGroup_Name]	Defines a descriptive name, which is used when associating the row in other tables. The valid value is a string of up to 40 characters. **Note:** ■ Each row must be configured with a unique name. ■ The parameter value cannot contain a forward slash (/).

Parameter	Description
'Topology Location' topology-location [IPGroup_ TopologyLocation]	Defines the display location of the IP Group in the Topology view of the Web interface. ■ [0] **Down** = (Default) The IP Group element is displayed on the lower border of the view. ■ [1] **Up** = The IP Group element is displayed on the upper border of the view. For more information on the Topology view, see Building and Viewing SIP Entities in Topology View.
'Type' type [IPGroup_Type]	Defines the type of IP Group. ■ [0] **Server** = Applicable when the destination address of the IP Group (e.g., ITSP, Proxy, IP-PBX, or Application server) is known. The address is configured by the Proxy Set that is associated with the IP Group. ■ [1] **User** = Represents a group of users such as IP phones and softphones where their location is dynamically obtained by the device when REGISTER requests and responses traverse (or are terminated) by the device. These users are considered remote (far-end). Typically, this IP Group is configured with a Serving IP Group that represents an IP-PBX, Application or Proxy server that serves this User-type IP Group. Each SIP request sent by a user of this IP Group is proxied to the Serving IP Group. For registrations, the device updates its registration database with the AOR and contacts of the users. Digest authentication using SIP 401/407 responses (if needed) is performed by the Serving IP Group. The device forwards these responses directly to the SIP users. To route a call to a registered user, a rule must be configured in the SBC IP-to-IP Routing table. The device searches the dynamic database (by using the Request-URI) for an entry that matches a registered AOR or Contact. Once an entry is found, the IP destination is obtained from this entry and a SIP request is sent to the destination. The device also supports NAT traversal for the SIP clients located behind NAT. In this case, the device must be defined with a global IP address.

Parameter	Description
	■ [2] **Gateway** = In scenarios where the device receives requests to and from a gateway representing multiple users. This IP Group type is necessary for any of the following scenarios: ✓ The IP Group cannot be defined as a Server-type since its address is initially unknown and therefore, a Proxy Set cannot be configured for it. ✓ The IP Group cannot be defined as a User-type since the SIP Contact header of the incoming REGISTER does not represent a specific user. The Request-URI user part can change and therefore, the device is unable to identify an already registered user and therefore, adds an additional record to the database. The IP address of the Gateway-type IP Group is obtained dynamically from the host part of the Contact header in the REGISTER request received from the IP Group. Therefore, routing to this IP Group is possible only once a REGISTER request is received (i.e., IP Group is registered with the device). If a REGISTER refresh request arrives, the device updates the new location (i.e., IP address) of the IP Group. If the REGISTER fails, no update is performed. If an UN-REGISTER request arrives, the IP address associated with the IP Group is deleted and therefore, no routing to the IP Group is done. You can view the registration status of the Gateway-type IP Group in the 'GW Group Registered Status' field, and view the IP address of the IP Group in the 'GW Group Registered IP Address' field if it is registered with the device.
'Proxy Set' `proxy-set-name` [IPGroup_ProxySetName]	Assigns a Proxy Set to the IP Group. All INVITE messages destined to the IP Group are sent to the IP address configured for the Proxy Set. To configure Proxy Sets, see Configuring Proxy Sets. **Note:** ■ The Proxy Set must be associated with the same SRD as that assigned to the IP Group. ■ You can assign the same Proxy Set to multiple IP Groups.

Parameter	Description
	■ Proxy Sets are used for Server-type IP Groups, but may in certain scenarios also be used for User-type IP Groups. For example, this is required in deployments where the device mediates between an IP PBX and a SIP Trunk, and the SIP Trunk requires SIP registration for each user that requires service. In such a scenario, the device must register all the users to the SIP Trunk on behalf of the IP PBX. This is done by using the User Information table, where each user is associated with the source IP Group (i.e., the IP PBX). To configure the User Information table, see SBC User Information for SBC User Database.
'IP Profile' `ip-profile-name` [IPGroup_ProfileName]	Assigns an IP Profile to the IP Group. By default, no value is defined. To configure IP Profiles, see Configuring IP Profiles.
'Media Realm' `media-realm-name` [IPGroup_MediaRealm]	Assigns a Media Realm to the IP Group. The Media Realm determines the UDP port range and maximum sessions on a specific IP interface for media traffic associated with the IP Group. By default, no value is defined. To configure Media Realms, see Configuring Media Realms. **Note:** If you delete a Media Realm in the Media Realms table that is assigned to the IP Group, the parameter value reverts to undefined.
'Internal Media Realm' `internal-media-realm-name` [IPGroup_ InternalMediaRealm]	Assigns an "internal" Media Realm to the IP Group. This is applicable when the device is deployed in a Microsoft Teams environment. The device selects this Media Realm (instead of the Media Realm assigned by the 'Media Realm' parameter above) if the value of the X-MS-UserLocation header in the incoming SIP message is "Internal" and the 'Teams Local Media Optimization Handling' parameter (see below) is configured to any value other than **None**. The Media Realm determines the UDP port range and maximum sessions on a specific IP interface for media traffic associated with the IP Group. By default, no value is defined. To configure Media Realms, see Configuring Media Realms. **Note:** ■ The parameter is applicable only if you have configured

Parameter	Description
	the 'Teams Local Media Optimization Handling' parameter (see below) to any value other than **None**. ■ If you delete a Media Realm in the Media Realms table that is assigned to the IP Group, the parameter value reverts to undefined. ■ If you don't configure the parameter, the device uses the Media Realm that you assigned by the 'Media Realm' parameter.
'Contact User' `contact-user` [IPGroup_ContactUser]	Defines the user part of the From, To, and Contact headers of SIP REGISTER messages, and the user part of the Contact header of INVITE messages received from this IP Group and forwarded by the device to another IP Group. The valid value is a string of up to 60 characters. By default, no value is defined. **Note:** ■ The parameter is applicable only to Server-type IP Groups. ■ The parameter is overridden by the 'Contact User' parameter in the Accounts table (see Configuring Registration Accounts).
'SIP Group Name' `sip-group-name` [IPGroup_SIPGroupName]	Defines the hostname (e.g., 194.90.179.0) which the device uses to overwrite the original hostname (host part) of the URI in certain SIP headers. Therefore, the parameter allows you to implement topology hiding in SIP messages, by concealing the hostname of the communicating UAs from each another. The affected SIP headers depend on whether the IP Group is the destination or source of the call: ■ **Destination IP Group:** The device overwrites the host part of the following SIP headers for messages sent (outgoing) to this IP Group: ✓ For all requests: Request-URI header (if the destination of the request isn't a registered user or Trunk Group, and the URL in the Request-URI is not GRUU) and P-Called-Party-ID header. ✓ For all non-REGISTER requests (e.g., INVITE and SUBSCRIBE): To header and Remote-Party-ID header (only the first Remote-Party-ID header whose type

Parameter	Description
	is "called" in the message). ✓ For INVITE requests only: If the 'Destination URI Input' parameter is configured for the source IP Group, the header type configured by the 'Destination URI Input' parameter is also modified according to the 'SIP Group Name' parameter of the destination IP Group. ■ **Source IP Group:** The device overwrites the host part of the following SIP headers for messages received (incoming) from this IP Group. ✓ For all types of requests: From header. ✓ For REGISTER requests: To header. ✓ For all non-REGISTER requests (e.g., INVITE and SUBSCRIBE): P-Preferred-Identity (only first P-Preferred-Identity header in message), P-Asserted-Identity (only first P-Asserted-Identity header in message), Remote-Party-ID (only the first Remote-Party-ID header whose type is "calling" in the message). ✓ For INVITE requests only: If the 'Source URI Input' parameter is configured for the source IP Group, the header type configured by the 'Source URI Input' parameter is also overwritten according to the 'SIP Group Name' parameter of the source IP Group. The valid value is a string of up to 100 characters. By default, no value is defined. **Note:** ■ When the IP Group is the source of the call, if you configure the destination IP Group's 'SIP Source Host Name' parameter (see below), the device ignores the 'SIP Group Name' parameter of the source IP Group. (The 'SIP Source Host Name' parameter also defines a URI host part to overwrite the original source host part, but it affects many more source-related SIP headers.) ■ When the parameter is configured for the source or destination IP Group, it overrides Inbound Message Manipulation rules (assigned by the 'Inbound Message Manipulation Set' parameter to the source IP Group)

Parameter	Description
	that manipulate the host part in the Request-URI, To, and From SIP headers. If you configure the parameter and you want to manipulate the host part in any of these SIP headers, assign your Message Manipulation rules to the destination IP Group using the 'Outbound Message Manipulation Set' parameter.
'Created By Routing Server' [IPGroup_ CreatedByRoutingServer]	(Read-only) Indicates whether the IP Group was created by a third-party routing server: ■ [0] **No** ■ [1] **Yes** For more information on the third-party routing server feature, see Centralized Third-Party Routing Server.
'Used By Routing Server' `used-by-routing-server` [IPGroup_ UsedByRoutingServer]	Enables the IP Group to be used by a third-party routing server for call routing decisions. ■ [0] **Not Used** (default) ■ [1] **Used** For more information on the third-party routing server feature, see Centralized Third-Party Routing Server.
'Proxy Set Connectivity' `show voip proxy sets status` [IPGroup_ ProxySetConnectivity]	(Read-only field) Displays the connectivity status with Server-type IP Groups. As the Proxy Set defines the address of the IP Group, the connectivity check (keep-alive) by the device is done to this address. ■ "NA": Functionality is not applicable due to one of the following: ✓ User-type IP Group. ✓ Server-type IP Group, but the keep-alive mechanism of its' associated Proxy Set is disabled. ■ "Not Connected": Keep-alive failure (i.e., no connectivity with the IP Group). ■ "Connected": Keep-alive success (i.e., connectivity with the IP Group). The connectivity status is also displayed in the Topology View page (see Building and Viewing SIP Entities in Topology View). **Note:**

Parameter	Description
	■ The feature is applicable only to Server-type IP Groups. ■ To support the feature, you must enable the keep-alive mechanism of the Proxy Set that is associated with the IP Group (see Configuring Proxy Sets). ■ If the Proxy Set is configured with multiple proxies (addresses) and at least one of them is "alive", the displayed status is "Connected". To view the connected proxy server, see Viewing Proxy Set Status. ■ The "Connected" status also applies to scenarios where the device rejects calls with the IP Group due to low QoE (e.g., low MOS), despite connectivity.
SBC General	
'Classify By Proxy Set' `classify-by-proxy-set` [IPGroup_ ClassifyByProxySet]	Enables classification of incoming SIP dialogs (INVITEs) to Server-type IP Groups based on Proxy Set (assigned using the IPGroup_ProxySetName parameter). ■ [0] **Disable** ■ [1] **Enable** = (Default) The device searches the Proxy Sets table for a Proxy Set that is configured with the same source IP address as that of the incoming INVITE (if host name, then according to the dynamically resolved IP address list). If such a Proxy Set is found, the device classifies the INVITE as belonging to the IP Group associated with the Proxy Set. **Note:** ■ The parameter is applicable only to Server-type IP Groups. ■ For security, it is recommended to classify SIP dialogs based on Proxy Set only if the IP address of the IP Group is unknown. In other words, if the Proxy Set associated with the IP Group is configured with an FQDN. In such cases, the device classifies incoming SIP dialogs to the IP Group based on the DNS-resolved IP address. If the IP address is known, it is recommended to use a Classification rule instead (and disable the Classify by Proxy Set feature), where the rule is configured with not only the IP address, but also with SIP message characteristics to increase the strictness of the classification process (see Configuring Classification

Parameter	Description
	Rules). The reason for preferring classification based on Proxy Set when the IP address is unknown is that IP address forgery (commonly known as IP spoofing) is more difficult than malicious SIP message tampering and therefore, using a Classification rule without an IP address offers a weaker form of security. When classification is based on Proxy Set, the Classification table for the specific IP Group is ignored. ■ If you have assigned the same Proxy Set to multiple IP Groups, disable the parameter and instead, use Classification rules to classify incoming SIP dialogs to these IP Groups. If the parameter is enabled, the device is unable to correctly classify incoming INVITEs to their appropriate IP Groups. ■ Classification by Proxy Set occurs only if classification based on the device's registration database fails (i.e., the INVITE is not from a registered user).
'SBC Operation Mode' `sbc-operation-mode` [IPGroup_SBCOperationMode]	Defines the device's operational mode for the IP Group. ■ [-1] **Not Configured** = (Default) ■ [0] **B2BUA** = Device operates as a back-to-back user agent (B2BUA), changing the call identifiers and headers between the inbound and outbound legs. ■ [1] **Call Stateful Proxy** = Device operates as a Stateful Proxy, passing the SIP message transparently between inbound and outbound legs. In other words, the same SIP dialog identifiers (tags, Call-Id and CSeq) occur on both legs (as long as no other configuration disrupts the CSeq compatibleness). ■ For more information on B2BUA and Stateful Proxy modes, see B2BUA and Stateful Proxy Operating Modes. **Note:** ■ If configured, the parameter overrides the 'SBC Operation Mode' parameter in the SRDs table.
'SBC Client Forking Mode' `sbc-client-forking-`	Defines call forking of INVITE messages to up to five separate SIP outgoing legs for User-type IP Groups. This

Parameter	Description
`mode` [IPGroup_ EnableSBCClientForking]	occurs if multiple contacts are registered under the same AOR in the device's registration database. ■ [0] **Sequential** = (Default) Sequentially sends the INVITE to each contact. If there is no answer from the first contact, it sends the INVITE to the second contact, and so on until a contact answers. If no contact answers, the call fails or is routed to an alternative destination, if configured. ■ [1] **Parallel** = Sends the INVITE simultaneously to all contacts. The call is established with the first contact that answers. ■ [2] **Sequential Available Only** = Sequentially sends the INVITE only to available contacts (i.e., not busy). If there is no answer from the first available contact, it sends the INVITE to the second contact, and so on until a contact answers. If no contact answers, the call fails or is routed to an alternative destination, if configured. **Note:** ■ The device can also fork INVITE messages received for a Request-URI of a specific contact (user) registered in the database to all other users located under the same AOR as the specific contact. This is configured by the [SBCSendInviteToAllContacts] parameter.
'CAC Profile' `cac-profile` [IPGroup_AdmissionProfile]	Assigns a Call Admission Control Profile (CAC rules) to the IP Group. By default, no value is defined. To configure CAC Profiles, see Configuring Call Admission Control on page 696.
'SIP Source Host Name' `sip-source-host-name` [IPGroup_ SIPSourceHostName]	Defines a hostname, which the device uses to overwrite the hostname of the URI in certain SIP headers. The parameter allows you to implement topology hiding for the source of SIP messages, by concealing the hostname of the source UA. The valid value is a string of up to 100 characters. By default, no value is defined. When the device forwards a SIP message to this IP Group, the configured hostname overwrites the host part in SIP headers (see below) that are concerned with the source of

Parameter	Description
	the message: ■ From, P-Asserted-Identity, P-Preferred-Identity, Referred-By, P-Charge-Info, Remote-Party-ID, P-Associated-URI, Diversion, and History-info headers. ■ If you configure the global parameter 'SIP Topology Hiding Mode' parameter to **Fallback to IP Addresses** and the 'Remote REFER Mode' [IpProfile_SBCRemoteReferBehavior] parameter to **Regular** (default), the host part in the Refer-To header is also overwritten. ■ For REGISTER requests, the host part in the To header is also overwritten. **Note:** ■ The parameter is applicable only when the IP Group is the destination of the call (not source). ■ This parameter has higher priority than the 'SIP Group Name' parameter (see above) of the source IP Group. When this parameter is configured, the device ignores the value of the 'SIP Group Name' parameter that is configured for the source IP Group. ■ The parameter is applicable only to SIP dialog-initiating requests and in-dialog REFER requests.
Advanced	
'Local Host Name' `local-host-name` [IPGroup_ContactName]	Defines the host name (string) that the device uses in the SIP message's Via and Contact headers. This is typically used to define an FQDN as the host name. The device uses this string for Via and Contact headers in outgoing INVITE messages sent to a specific IP Group, and the Contact header in SIP 18x and 200 OK responses for incoming INVITE messages received from a specific IP Group. The IP-to-Tel Routing table can be used to identify the source IP Group from where the INVITE message was received. If the parameter is not configured, these headers are populated with the device's dotted-decimal IP address of the network interface on which the message is sent. By default, no value is defined. **Note:** To ensure proper device handling, the parameter should be a valid FQDN.

Parameter	Description
'UUI Format' uui-format [IPGroup_UUIFormat]	Enables the generation of the Avaya UCID value, adding it to the outgoing INVITE sent to this IP Group. ■ [0] **Disabled** (default) ■ [1] **Enabled** This provides support for interworking with Avaya equipment by generating Avaya's UCID value in outgoing INVITE messages sent to Avaya's network. The device adds the UCID in the User-to-User SIP header. Avaya's UCID value has the following format (in hexadecimal): 00 + FA + 08 + node ID (2 bytes) + sequence number (2 bytes) + timestamp (4 bytes) This is interworked in to the SIP header as follows: User-to-User: 00FA080019001038F725B3;encoding=hex **Note:** To define the Network Node Identifier of the device for Avaya UCID, use the 'Network Node ID' (NetworkNodeId) parameter.
'Always Use Src Address' always-use-source-addr [IPGroup_ AlwaysUseSourceAddr]	Enables the device to always send SIP requests and responses, within a SIP dialog, to the source IP address received in the previous SIP message packet. This feature is especially useful in scenarios where the IP Group endpoints are located behind a NAT firewall (and the device is unable to identify this using its regular NAT mechanism). ■ [0] **No** = (Default) The device sends SIP requests according to the settings of the global parameter, SIPNatDetection. ■ [1] **Yes** = The device sends SIP requests and responses to the source IP address received in the previous SIP message packet. For more information on NAT traversal, see Remote UA behind NAT.
SBC Advanced	
'Source URI Input' src-uri-input [IPGroup_SourceUriInput]	Defines the SIP header in the incoming INVITE that is used for call matching characteristics based on source URIs. ■ [-1] **Not Configured** (default) ■ [0] **From**

Parameter	Description
	■ [1] **To** ■ [2] **Request-URI** ■ [3] **P-Asserted - First Header** ■ [4] **P-Asserted - Second Header** ■ [5] **P-Preferred** ■ [6] **Route** ■ [7] **Diversion** ■ [8] **P-Associated-URI** ■ [9] **P-Called-Party-ID** ■ [10] **Contact** ■ [11] **Referred-by** **Note:** ■ The parameter is applicable only when classification is done according to the Classification table (see Configuring Classification Rules on page 703). ■ Once classified, the device uses the URI of the selected header for the following SIP headers in the outgoing INVITE: From, P-Asserted (if exists), P-Preferred (if exists), and Remote-Party-ID (if exists). ■ If the configured SIP header does not exist in the incoming INVITE message, the classification of the message to a source IP Group fails. ■ If the device receives an INVITE as a result of a REFER request or a 3xx response, then the incoming INVITE is routed according to the Request-URI. The device identifies such INVITEs according to a specific prefix in the Request-URI header, configured by the SBCXferPrefix parameter. Therefore, in this scenario, the device ignores the parameter setting.
'Destination URI Input' `dst-uri-input` [IPGroup_DestUriInput]	Defines the SIP header in the incoming INVITE to use as a call matching characteristic based on destination URIs. The parameter is used for classification and routing purposes. The device first uses the parameter's settings as a matching characteristic (input) to classify the incoming INVITE to an IP Group (source IP Group) in the Classification table. Once

Parameter	Description
	classified, the device uses the parameter for routing the call. For example, if set to To, the URI in the To header of the incoming INVITE is used as a matching characteristic for classifying the call to an IP Group in the Classification table. Once classified, the device uses the URI in the To header as the destination. ■ [-1] **Not Configured** (default) ■ [0] **From** ■ [1] **To** ■ [2] **Request-URI** ■ [3] **P-Asserted - First Header** ■ [4] **P-Asserted - Second Header** ■ [5] **P-Preferred** ■ [6] **Route** ■ [7] **Diversion** ■ [8] **P-Associated-URI** ■ [9] **P-Called-Party-ID** ■ [10] **Contact** ■ [11] **Referred-By** **Note:** ■ The parameter can be configured for an IP Group regardless of the way in which SIP requests are classified to the IP Group (by Classification table, by Proxy Set or by the Registration database). ■ The Request-URI in the outbound side is modified according to the header selected by this parameter (if this parameter is configured), unless the Request-URI is overridden again by some other feature (e.g., Outbound Message Manipulations). ■ If the configured SIP header does not exist in the incoming INVITE message, the classification of the message to a source IP Group fails. ■ If the device receives an INVITE as a result of a REFER request or a 3xx response, the incoming INVITE is routed according to the Request-URI. The device identifies such

Parameter	Description
	INVITEs according to a specific prefix in the Request-URI header, configured by the SBCXferPrefix parameter. Therefore, in this scenario, the device ignores the parameter setting.
'SIP Connect' `sip-connect` [IPGroup_SIPConnect]	Defines the IP Group as representing multiple registering servers, each of which may use a single registration, yet represent multiple users. In addition, it defines how the device saves registration information for REGISTER messages received from the IP Group, in its registration database. For requests routed to the IP Group's users, the device replaces the Request-URI header with the incoming To header (which contains the remote phone number). ■ [0] **No** = (Default) Disables the SIP Connect feature. No extra key based on source IP address is added to the registration database and registration is done by Contact and Address of Record (AoR). ■ [1] **Yes** = Enables the SIP Connect feature. For initial registrations that are received from the IP Group, the device attempts to add two *keys* representing the user to its registration database: ✓ Key 1: The first key contains the incoming REGISTER message's source IP address, port (only if UDP), and SIP Interface ID (e.g., "10.33.3.3:5010#1"). ✓ Key 2: The second key contains the incoming REGISTER message's URI (user@host) of the Contact header, source IP address, port (only if UDP), and SIP Interface ID (e.g., "user@host.com#10.33.3.3:5010#1"). The device classifies incoming non-REGISTER SIP dialog requests (e.g., INVITEs) from this IP Group, by first using the regular user search method in the registration database by Contact-AoR pair matching. If unsuccessful, the device searches the registration database for a matching Key 2 (i.e., Contact URI, source IP address, and port if the transport type is UDP). If no matching Key 2 exists, the device then searches for a matching Key 1 (i.e., source IP address only and port if the transport type is UDP). If no key is found at all, the device continues with the next Classification stage (e.g., by Proxy Set). **Note:**

Parameter	Description
	■ The parameter is applicable only to User-type IP Groups. ■ The parameter is applicable only to the SBC application.
'SBC PSAP Mode' `sbc-psap-mode` [IPGroup_SBCPSAPMode]	Enables E9-1-1 emergency call routing in a Microsoft Skype for Business environment. ■ [0] **Disable** (default) ■ [1] **Enable** For more information, see E9-1-1 Support for Microsoft Skype for Business.
'Route Using Request URI Port' `use-requri-port` [IPGroup_ SBCRouteUsingRequestURI Port]	Enables the device to use the port indicated in the Request-URI of the incoming message as the destination port when routing the message to the IP Group. The device uses the IP address (and not port) that is configured for the Proxy Set associated with the IP Group. The parameter thus allows the device to route calls to the same server (IP Group), but different port. ■ [0] **Disable** = (Default) The port configured for the associated Proxy Set is used as the destination port. ■ [1] **Enable** = The port indicated in the Request-URI of the incoming message is used as the destination port.
'Media TLS Context' `dtls-context` [IPGroup_DTLSContext]	Assigns a TLS Context (TLS configuration) to the IP Group that is used for secured media sessions (e.g., DTLS and MSRPS) with the IP Group. The default is the default TLS Context ("default" at Index 0). To configure TLS Contexts, see Configuring TLS Certificates on page 158. For more information on DTLS, see SRTP using DTLS Protocol on page 233. For more information on MSRP, see Configuring Message Session Relay Protocol on page 779.
'Keep Original Call-ID' `sbc-keep-call-id` [IPGroup_ SBCKeepOriginalCallID]	Enables the device to use the same call identification (SIP Call-ID header value) received in incoming messages for the call identification in outgoing messages. The call identification value is contained in the SIP Call-ID header. ■ [0] **No** = (Default) The device creates a new Call-ID value for the outgoing message.

Parameter	Description
	■ [1] **Yes** = The device uses the same Call-ID value received in the incoming message for the Call-ID in the outgoing message. **Note:** ■ When the device sends an INVITE as a result of a REFER/3xx termination, the device always creates a new Call-ID value and ignores the parameter's settings.
'Dial Plan' `sbc-dial-plan-name` [IPGroup_ SBCDialPlanName]	Assigns a Dial Plan to the IP Group. The device searches the Dial Plan for a dial plan rule that matches the prefix of the source number and if not found, for a rule that matches the prefix of the destination number. If a matching Dial Plan rule is found, the rule's tag is used in the routing or manipulation processes as source or destination tags. To configure Dial Plans, see Configuring Dial Plans. **Note:** ■ For IP-to-IP Routing rules that are configured for destination based on tags (i.e., 'Destination Type' parameter configured to Destination Tag), the parameter is applicable only to the source IP Group and the device searches the Dial Plan for a dial plan rule that matches the prefix of the destination number only. For more information on routing based on destination tags, see Using Dial Plan Tags for Routing Destinations.
'Call Setup Rules Set ID' `call-setup-rules-set-id` [IPGroup_ CallSetupRulesSetId]	Assigns a Call Setup Rule Set ID to the IP Group. The device runs the Call Setup rule immediately before the routing stage (i.e., only after the classification and manipulation stages). By default, no value is assigned. To configure Call Setup Rules, see Configuring Call Setup Rules. **Note:** ■ Call Setup Rules that are triggered from the IP Groups table (incoming IP Group) are done after identifying the incoming SIP Interface and after classification and manipulation for identifying the incoming IP Group, but before the routing stage (IP-to-IP Routing table). This supports all types of queries (Dial Plan, LDAP, ENUM, and HTTP).

Parameter	Description
'Tags' tags [IPGroup_Tags]	Defines a tag, which can be implemented in one of the following manners: ■ **Classification based on source tags:** If the tag (name=value or value only) is the same tag as that of the incoming SIP dialog (obtained from the Call Setup Rule associated with the SIP Interface on which the dialog is received) and configured in the Classification table, then the incoming dialog is classified to this IP Group. For more information, see Configuring Classification Based on Tags on page 714. ■ **Routing based on destination tags:** Assigns a Dial Plan tag which determines whether the incoming SIP dialog is sent to this IP Group. The parameter is used when IP-to-IP Routing rules are configured for destinations-based on tags (i.e., 'Destination Type' parameter configured to **Destination Tag**). For more information, see Using Dial Plan Tags for Routing Destinations. The valid value is a string of up to 70 characters. You can configure the parameter with up to five tags, where each tag is separated by a semicolon (;). However, you can configure only up to four tags containing a name and value (e.g., Country=Ireland), and **one** tag containing a value only (e.g., Ireland). You can also configure multiple tags with the same name (e.g., Country=Ireland;Country=Scotland). The following example configures the maximum number of tags (i.e., four name=value tags and one value-only tag): Country=Ireland;Country=Scotland;Country=RSA;Country=Canada;USA. **Note:** ■ For tag-based classification, if multiple IP Groups are configured with the same tag, the device classifies the incoming SIP dialog to the first matching IP Group.
'SBC Alternative Routing Reasons Set' sbc-alt-route-reasons-set [IPGroup_ SBCAltRouteReasonsSetName]	Assigns an Alternative Reasons Set to the IP Group. This defines SIP response codes, which if received by the device from the IP Group, triggers alternative routing. Alternative routing could mean trying to send the SIP message to another online proxy (address) that is configured for the Proxy Set associated with the IP Group, or sending it to an alternative IP-to-IP Routing rule. For configuring Alternative Reasons Sets and for more information on how the device

Parameter	Description
	performs alternative routing, see Configuring SIP Response Codes for Alternative Routing Reasons on page 742. By default, no value is defined.
'Teams Local Media Optimization Handling' `teams-local-media-optimization-handling` [IPGroup_TeamsLocalMediaOptimization]	Enables and defines Local Media Optimization handling when the central SBC device (proxy SBC scenario) is deployed in a Microsoft Teams environment. The handling is based on supplementary information provided by Microsoft proprietary SIP headers, X-MS-UserLocation and X-MS-MediaPath. ■ [0] **None** = (Default) The device ignores the Teams headers in the SIP message and uses the "regular" Media Realm assigned by the IP Group's 'Media Realm' parameter for the call. ■ [1] **Teams Decides** = The device's call handling depends on the Teams headers and call direction: ✓ Teams-to-SBC Call: If the call is a **primary** route, the device uses the Teams headers (X-MS-UserLocation and X-MS-MediaPath) in the incoming INVITE message. If the X-MS-UserLocation header value is "internal", the device uses the Media Realm assigned by the IP Group's 'Internal Media Realm' parameter. If the X-MS-UserLocation header value is "external", the device uses the Media Realm assigned by the IP Group's 'Media Realm' parameter. Based on the X-MS-MediaPath header, the device determines if it's a direct or non-direct media call. If the X-MS-UserLocation header value is "internal" and the first value of the X-MS-MediaPath header is the same value as configured for the IP Group's 'Local Host Name' parameter, the call traverses the device; otherwise, it bypasses the device (direct media). The X-MS-UserLocation and X-MS-MediaPath headers can change upon re-INVITE messages (such as for conference calls). If the call is a **non-primary** route (e.g., alternative route, 3xx, or forking), the device only uses the X-MS-UserLocation header in the incoming INVITE message, which it uses to select the appropriate Media Realm (as explained previously for primary

Parameter	Description
	routes). For non-primary routes, the media traverses the device (i.e., no direct media). ✓ SBC-to-Teams Call: The device forwards the INVITE message with the SDP received from the peer side to Teams according to the configuration of the IP Group's 'Teams Local Media Optimization Initial Behavior' parameter (see below). Based on the Teams headers in the 200 OK response from Teams, the device selects the appropriate Media Realm (as explained previously for Teams-to-SBC calls) and determines if it's a direct or non-direct media call (as explained previously for Teams-to-SBC calls). ■ [2] **SBC Decides** = The device only uses the X-MS-UserLocation header in the SIP message, which it uses to select the appropriate Media Realm (as explained previously for the **Teams Decides** option). When configured to the **SBC Decides** option, the media traverses the device (i.e., no direct media). **Note:** For an outgoing INVITE message to Teams, the device sends the call as a non-direct media and uses the Media Realm assigned by the IP Group's 'Media Realm' parameter. For an overview of Microsoft Teams Local Media Optimization feature, see Microsoft Teams with Local Media Optimization on page 388.
'Teams Local Media Optimization Initial Behavior' `teams-local-mo-initial-behavior` [IPGroup_TeamsLocalMOInitialBehavior]	Defines how the central SBC device (proxy SBC scenario) initially sends the received INVITE message with the SDP Offer to Teams when the device is deployed in a Microsoft Teams environment for its Local Media Optimization feature. The parameter is applicable when the device receives the SDP Offer from a remote site SBC and the 'Teams Local Media Optimization Handling' parameter is configured to **Teams Decides** or **SBC Decides**. ■ [0] **Direct Media** = (Default) The device sends the SDP Offer as is to Teams, indicating that the call is intended to be a direct media call (i.e., doesn't traverse the device). **Note:** This option is applicable only when 'Teams Local Media Optimization Handling' parameter is configured to **Teams Decides**.

Parameter	Description
	■ [1] **Internal** = The device sends the SDP Offer using the internal Media Realm (see the IP Group's 'Internal Media Realm' parameter) to Teams, indicating that the call is intended to be a non-direct media call (i.e., media traverses the central SBC device). ■ [2] **External** = The device sends the SDP Offer using the external (regular) Media Realm (see the IP Group's 'Media Realm' parameter) to Teams, indicating that the call is intended to be a non-direct media call (i.e., media traverses the central SBC device). For a brief overview of Microsoft Teams Local Media Optimization feature, see Microsoft Teams with Local Media Optimization on page 388.
'Teams Direct Routing Mode' `teams-direct-routing-mode` [IPGroup_ TeamsDirectRoutingMode]	Enables the device to include Microsoft's proprietary X-MS-SBC header in outgoing SIP INVITE and OPTIONS messages in a Microsoft Teams Direct Routing environment. The header is used by Microsoft Teams to identify vendor equipment (e.g., AudioCodes SBC). ■ [0] **Disable** = (Default) The device doesn't include the header in the outgoing SIP message. ■ [1] **Enable** = The device includes the header in the outgoing SIP message. The header's value is in the format 'AudioCodes/<model>/<firmware>', where: ✓ *model* is the product name of your AudioCodes device (valid values are listed by Microsoft at https://docs.microsoft.com/en-us/microsoftteams/direct-routing-border-controllers). ✓ *firmware* is the software version running on the device. **Note:** ■ You can't modify or remove the header using Message Manipulation.
Quality of Experience	
'QoE Profile' `qoe-profile` [IPGroup_QOEProfile]	Assigns a Quality of Experience Profile rule. By default, no value is defined. To configure Quality of Experience Profiles, see Configuring

Parameter	Description
	Quality of Experience Profiles.
'Bandwidth Profile' bandwidth-profile [IPGroup_BWProfile]	Assigns a Bandwidth Profile rule. By default, no value is defined. To configure Bandwidth Profiles, see Configuring Bandwidth Profiles.
Message Manipulation	
'Inbound Message Manipulation Set' inbound-mesg-manipulation-set [IPGroup_InboundManSet]	Assigns a Message Manipulation Set (rule) to the IP Group for SIP message manipulation on the inbound leg. By default, no value is defined. To configure Message Manipulation rules, see Configuring SIP Message Manipulation. **Note:** ■ The IPGroup_SIPGroupName parameter overrides inbound message manipulation rules (assigned to the IPGroup_InboundManSet parameter) that manipulate the host name in Request-URI, To, and/or From SIP headers. If you want to manipulate the host name using message manipulation rules in any of these SIP headers, you must apply your manipulation rule (Manipulation Set ID) to the IP Group as an Outbound Message Manipulation Set (see the [IPGroup_OutboundManSet] parameter), when the IP Group is the destination of the call.
'Outbound Message Manipulation Set' outbound-mesg-manipulation-set [IPGroup_ OutboundManSet]	Assigns a Message Manipulation Set (rule) to the IP Group for SIP message manipulation on the outbound leg. By default, no value is defined. To configure Message Manipulation rules, see Configuring SIP Message Manipulation. **Note:** If you assign a Message Manipulation Set ID that includes rules for manipulating the host name in the Request-URI, To, and/or From SIP headers, the parameter overrides the [IPGroup_SIPGroupName] parameter.
'Message Manipulation User-Defined String 1' msg-man-user-defined-string1 [IPGroup_	Defines a value for the SIP user part that can be used in Message Manipulation rules configured in the Message Manipulations table. The Message Manipulation rule obtains this value from the IP Group, by using the following syntax:

Parameter	Description
MsgManUserDef1]	`param.ipg.<src|dst>.user-defined.<0>`. The valid value is a string of up to 30 characters. By default, no value is defined. To configure Message Manipulation rules, see Configuring SIP Message Manipulation.
'Message Manipulation User-Defined String 2' `msg-man-user-defined-string2` [IPGroup_ MsgManUserDef2]	Defines a value for the SIP user part that can be used in Message Manipulation rules configured in the Message Manipulations table. The Message Manipulation rule obtains this value from the IP Group, by using the following syntax: param.ipg.<src|dst>.user-defined.<1>. The valid value is a string of up to 30 characters. By default, no value is defined. To configure Message Manipulation rules, see Configuring SIP Message Manipulation.
'Proxy Keep-Alive using IP Group Settings' `proxy-keepalive-use-ipg` [ProxyKeepAliveUsingIPG]	Enables the device to apply certain IP Group settings to keep-alive SIP OPTIONS messages that are sent by the device to the proxy server. The parameter is applicable only if you have enabled proxy keep-alive for the Proxy Set that is associated with the IP Group (see Configuring Proxy Sets on page 483). ■ [0] **Disable** = (Default) The IP Group's settings are not applied to the SIP messages. ■ [1] **Enable** = The following IP Group settings are applied (if configured) to the proxy keep-alive SIP messages: ✓ The IP Group's 'SIP Group Name' parameter (see above) value is used in the SIP messages. ✓ The IP Group's 'Outbound Message Manipulation Set' parameter (see above) is applied to the SIP messages (instead of manipulations configured by the [GWOutboundManipulationSet] parameter). You can also use the manipulation syntax "param.ipg.dst" for denoting the IP Group's parameters. ✓ When filtering logs (configured in the Logging Filters table), the SIP messages are filtered by IP Group. For more information on log filtering, see Configuring Log Filter Rules on page 1059. **Note:** When multiple IP Groups are associated with the

Parameter	Description
	same Proxy Set, the parameter can be enabled only on **one** of them.
SBC Registration and Authentication	
'Max. Number of Registered Users' `max-num-of-reg-users` [IPGroup_ MaxNumOfRegUsers]	Defines the maximum number of users in this IP Group that can register with the device. The default is -1, meaning that no limitation exists for registered users. **Note:** The parameter is applicable only to User-type IP Groups.
'Registration Mode' `registration-mode` [IPGroup_ RegistrationMode]	Defines the registration mode for the IP Group. ■ [0] **User Initiates Registration** (default) ■ [1] **SBC Initiates Registration** = Used when the device serves as a client (e.g., with an IP PBX). This functions only with the User Information table (see Configuring SBC User Information Table through Web Interface on page 590). ■ [2] **Registrations not Needed** = The device adds users to its database in active state.
'User Stickiness' `sbc-user-stickiness` [IPGroup_ SBCUserStickiness]	Enables user "stickiness" (binding) to a specific registrar server. The registrar server is one of the IP addresses of the Proxy Set associated with this Server-type IP Group. This feature applies to users belonging to a User-type IP Group that are routed to this destination Server-type IP Group. ■ [0] **Disable** = After a successful initial registration of the user to a registrar, whenever the device receives a SIP request or registration refresh from the user, the device sends the request to whichever registrar (IP address of the Proxy Set) is currently active. In the case of proxy load-balancing, there is no certainty to which IP address the request is routed. ■ [1] **Enable** = The device always routes SIP requests (INVITEs, SUBSCRIBEs and REGISTER refreshes) received from the user to the same registrar server to which the last successful REGISTER request for that user was routed. In other words, once initial registration of the user to one of the IP addresses of the Proxy Set associated with this destination Server-type IP Group is

Parameter	Description
	successful (i.e., 200 OK), binding occurs to this specific address (registrar) and all future SIP requests from the user are routed (based on matched routing rule) only to this specific registrar. **Note:** ■ The parameter is applicable only to Server-type IP Groups ■ The Proxy Set associated with the Server-type IP Group must be configured with multiple IP addresses (or an FQDN that resolves into multiple IP addresses). ■ This feature is also applicable to IP Group Sets (see Configuring IP Group Sets). If a user is bound to a registrar associated with this Server-type IP Group which also belongs to an IP Group Set, IP Group Set logic of choosing an IP Group is ignored and instead, the device always routes requests from this user to this specific registrar. ■ A user's "stickiness" to a specific registrar ends upon the following scenarios: ✓ If you modify the Proxy Set. ✓ If the Proxy Set is configured with an FQDN and a DNS resolution refresh removes the IP address to which the user is bound. ✓ User registration expires or the user initiates an unregister request. ■ The Proxy Set's Hot-Swap feature (for proxy redundancy) is not supported for users that are already bound to a registrar. However, you can achieve proxy "hot-swap" for failed initial (non-bounded) REGISTER requests. If the device receives a failure response for the initial REGISTER request and you have configured this response code for the Alternative Reasons Set associated (by the 'SBC Alternative Routing Reasons Set' parameter below) with the IP Group (see Configuring SIP Response Codes for Alternative Routing Reasons), "hot-swap" to the other IP addresses of the Proxy Set is done until a success response is received from one of the addresses. For failed REGISTER refresh requests from users that are already bound to a registrar, no "hot-

Parameter	Description
	swap" occurs for that request; only for subsequent refresh requests. ■ When using the User Information table (see SBC User Information for SBC User Database), registrar "stickiness" is supported only when the user initiates the REGISTER request. Therefore, you must configure the 'Registration Mode' parameter of the IP Group (User-type) to which the user belongs, to **User Initiates Registration**. ■ This feature is also supported when the device operates in HA mode; registrar "stickiness" is retained even after an HA switchover.
'User UDP Port Assignment' `user-udp-port-assignment` [IPGroup_ UserUDPPortAssignment]	Enables the device to assign a unique, local UDP port (for SIP signaling) per registered user (User-type IP Group) on the leg interfacing with the proxy server (Server-type IP Group). The port is used for incoming (from the proxy to the user) and outgoing (from the user to the proxy) SIP messages. Therefore, the parameter must be enabled for the IP Group of the proxy server. ■ [0] **Disable** = (Default) The device uses the same local UDP port for all the registered users. This single port is configured for the SIP Interface ('UDP Port' parameter) associated with the Proxy Set of the proxy server. ■ [1] **Enable** = The device assigns each registered user a unique local port, chosen from a configured UDP port range. The port range is configured for the SIP Interface ('Additional UDP Ports' parameter) associated with the proxy server. The device assigns a unique port upon the first REGISTER request received from the user. Subsequent SIP messages other than REGISTER messages (e.g., INVITE) from the user are sent to the proxy server on this unique local port. The device rejects the SIP request if there is no available unique port for use (due to the number of registered users exceeding the configured port range). The same unique port is also used for registration refreshes. The device de-allocates the port for registration expiry. For SIP requests from the proxy server, the local port on which they are received is irrelevant (unique port or any other port); the device does not use

Parameter	Description
	this port to identify the registered user. **Note:** ■ This feature does not apply to SIP requests received from non-registered users. For these users, the device sends all requests to the proxy server on the single port configured for the SIP Interface ('UDP Port' parameter). ■ For HA systems, the unique port assigned to a registered user is also used after an HA switchover. ■ This feature is applicable only if the user initiates registration (i.e., user sends the REGISTER request). In other words, the 'Registration Mode' parameter of the IP Group of the user must be configured to **User Initiates Registration**.
'Authentication Mode' `authentication-mode` [IPGroup_ AuthenticationMode]	Defines the authentication mode. ■ [0] **User Authenticates** = (Default) The device does not handle authentication, but simply forwards the authentication messages between the SIP user agents. ■ [1] **SBC as Client** = The device authenticates as a client. It receives the 401/407 response from the proxy requesting for authentication. The device sends the proxy the authorization credentials (i.e., username and password), which is obtained from one of the following (in chronological order): **a.** If an Account exists in the Accounts table (see Configuring Registration Accounts) for the Served IP Group and the Serving IP Group (i.e., the IP Group you are now configuring), the device uses the user-name and password configured for the Account (only if authenticating a Server-type IP Group). **b.** User Information file (see Configuring User Information on page 589). **c.** The device uses the username and password configured for this IP Group in the IP Groups table ('Username' and 'Password' parameters). **d.** Global username and password parameters (only if authenticating a Server-type IP Group). **e.** Sends a request to users requesting credentials

Parameter	Description
	(only if authenticating a User-type IP Group). ■ [2] **SBC as Server** = The device acts as an Authentication server: ✓ Authenticates SIP clients. This is applicable only to User-type IP Groups. The device authenticates incoming SIP requests from users belonging to this IP Group. The device authenticates all the users, using the username and password configured in the IP Group's 'Username' and 'Password' parameters. However, if a user appears in the User Information table and which is configured with a username and password, then the device authenticates the user with the credentials in the User Information table (see Configuring SBC User Information on page 590). If you have not configured any username and password, and the [SBCServerAuthMode] parameter is configured to [0] (default), the device rejects the incoming SIP request. ✓ Authenticates SIP servers. This is applicable only to Server-type IP Groups.
'Authentication Method List' `authentication-method-list` [IPGroup_MethodList]	Defines SIP methods received from the IP Group that must be challenged by the device when the device acts as an Authentication server. If no methods are configured, the device doesn't challenge any methods. By default, no value is defined. To define multiple SIP methods, use the backslash (\) to separate each method (e.g., INVITE\REGISTER). To authenticate only setup INVITE requests (and not re-INVITE requests), configure the parameter to "setup-invite" (without quotation marks). **Note:** The parameter is applicable only if the 'Authentication Mode' parameter is set to **SBC as Server**.
'SBC Server Authentication Type' `sbc-server-auth-type` [IPGroup_ TypeSBCServerAuthType]	Defines the authentication method when the device, as an Authentication server, authenticates SIP requests from the IP Group. ■ [-1] **According to Global Parameter** = (Default) Authentication is according to the settings of the SBCServerAuthMode parameter. ■ [0] **Authentication is performed locally** = The device authenticates incoming SIP requests locally. For more

Parameter	Description
	information, see SIP Authentication Server Functionality on page 678. ■ [2] **According to draft-sterman-aaa-sip-01** = The device authenticates incoming SIP requests using a remote RADIUS server, based on Internet Draft "draft-sterman-aaa-sip-01". For more information, see RADIUS-based User Authentication on page 679. ■ [3] **Authenticate with OAuth authorization** = The device authenticates incoming SIP requests according to token-based authentication with an OAuth2 Authorization server. The Authentication server is configured as a Remote Web Service and is assigned to the IP Group using the IP Group's 'OAuth HTTP Service' parameter. For more information, see OAuth2-based User Authentication on page 680. ■ [4] **ARM Authentication** = The device authenticates incoming SIP requests (INVITE or REGISTER) from User-type IP Groups, by first obtaining (REST-based API query) the user's password from a third-party routing server or AudioCodes ARM where the password is stored. Once the password is supplied, the device continues with the regular SIP digest authentication process (challenge) with the user. For more information on the third-party routing server or ARM, see Third-Party Routing Server or AudioCodes Routing Manager on page 328. **Note:** ■ If you configure the parameter to either **Authentication is performed locally**, **According to draft-sterman-aaa-sip-01**, **Authenticate with OAuth authorization**, or **ARM Authentication**, you also need to configure the IP Group's 'Authentication Mode' parameter to **SBC as Server.** ■ If you configure the parameter to **ARM Authentication**, you also need to configure the IP Group's 'Authentication Method List' parameter to authenticate INVITE or REGISTER messages.
'OAuth HTTP Service' `oauth-http-service` [IPGroup_	Assigns a Remote Web Service to the IP Group. The Remote Web Service represents the OAuth Authorization server, which the device uses to authenticate incoming SIP

Parameter	Description
OAuthHTTPService]	requests as a server. The device sends the OAuth token received from the client to the Authorization server for authentication. To configure Remote Web Services, see Configuring Remote Web Services on page 316. For more information on OAuth-based authentication, see OAuth2-based User Authentication on page 680. **Note:** The parameter is applicable only if the IP Group's 'SBC Server Authentication' parameter is configured to **Authenticate with OAuth authorization**.
'Username' `username` [IPGroup_Username]	Defines the shared username for authenticating the IP Group. The valid value is a string of up to 60 characters. By default, no username is defined. **Note:** If you configure the 'Authentication Mode' parameter to **SBC as Server**, you need to specify the SIP request (method) types (e.g., INVITE) that must be challenged by the device, using the IP Group's 'Authentication Method List' parameter.
'Password' `password` [IPGroup_Password]	Defines the shared password for authenticating the IP Group. The valid value is a string of up to 51 characters. By default, no password is defined. **Note:** ■ If you configure the 'Authentication Mode' parameter to **SBC as Server**, you can specify the SIP request (method) types (e.g., INVITE) that must be challenged by the device, using the IP Group's 'Authentication Method List' parameter. ■ The password cannot be configured with wide characters.
GW Group Status	
'GW Group Registered IP Address'	(Read-only field) Displays the IP address of the IP Group entity (gateway) if registered with the device; otherwise, the field is blank. **Note:** The field is applicable only to Gateway-type IP Groups (i.e., the 'Type' parameter is configured to Gateway).

Parameter	Description
'GW Group Registered Status'	(Read-only field) Displays whether the IP Group entity (gateway) is registered with the device ("Registered" or "Not Registered"). Note: The field is applicable only to Gateway-type IP Groups (i.e., the 'Type' parameter is configured to Gateway).

Configuring Proxy Sets

The Proxy Sets table lets you configure up to 625 Proxy Sets. A Proxy Set defines the address (IP address or FQDN) and transport type (e.g., UDP or TCP) of a SIP server (e.g., SIP proxy and SIP registrar server). The Proxy Set represents the destination of the IP Group configuration entity.

- You can configure each Proxy Set with up to 10 proxy servers (rows) in the Proxy Address table (a "child" of the Proxy Sets table), configured as IP addresses (in dotted-decimal notation) and/or DNS hostnames (FQDN).
- Each Proxy Set supports up to 15 DNS-resolved IP addresses.
- Each Proxy Set supports up to 15 IP addresses, regardless of how the IP address is obtained--DNS resolved or manually configured (dotted-decimal notation).
- For all Proxy Sets together, the device supports up to 4,500 DNS-resolved IP addresses. If the DNS resolution provides more than this number, it ignores the extra addresses.
- An SRV query sent by the device can return up to 50 hostnames. For each hostname, the subsequent DNS A-record query sent by the device can resolve into up to 50 IP addresses.

Multiple proxy servers enables you to implement proxy load balancing and redundancy. These features are supported by the device's proxy keep-alive feature, which when enabled, sends keep- alive messages (SIP OPTIONS) to all configured proxy servers to determine their connectivity status (offline or online). You can also configure the device to consider the proxy as offline if specific SIP response codes are received in response to the keep-alive messages. You can configure the number of required consecutive successful keep-alive messages before the device considers a previously offline proxy as online. This mechanism avoids the scenario in which the device falsely detects a proxy as being online when it is actually offline, resulting in call routing failure.

You can assign each Proxy Set a specific TLS Context (TLS configuration), enabling you to use different TLS settings (including certificates) per SIP entity (IP Group).

You can also enable the device to classify incoming SBC SIP dialogs to IP Groups, based on Proxy Set. If the source address of the incoming SIP dialog is the same as the address of a Proxy Set, the device classifies the SIP dialog as belonging to the IP Group that is associated with the Proxy Set.

To use a configured Proxy Set, you need to assign it to an IP Group in the IP Groups table (see Configuring IP Groups). When the device sends INVITE messages to an IP Group, it sends it to the address configured for the Proxy Set. You can assign the same Proxy Set to multiple IP Groups (belonging to the same SRD).

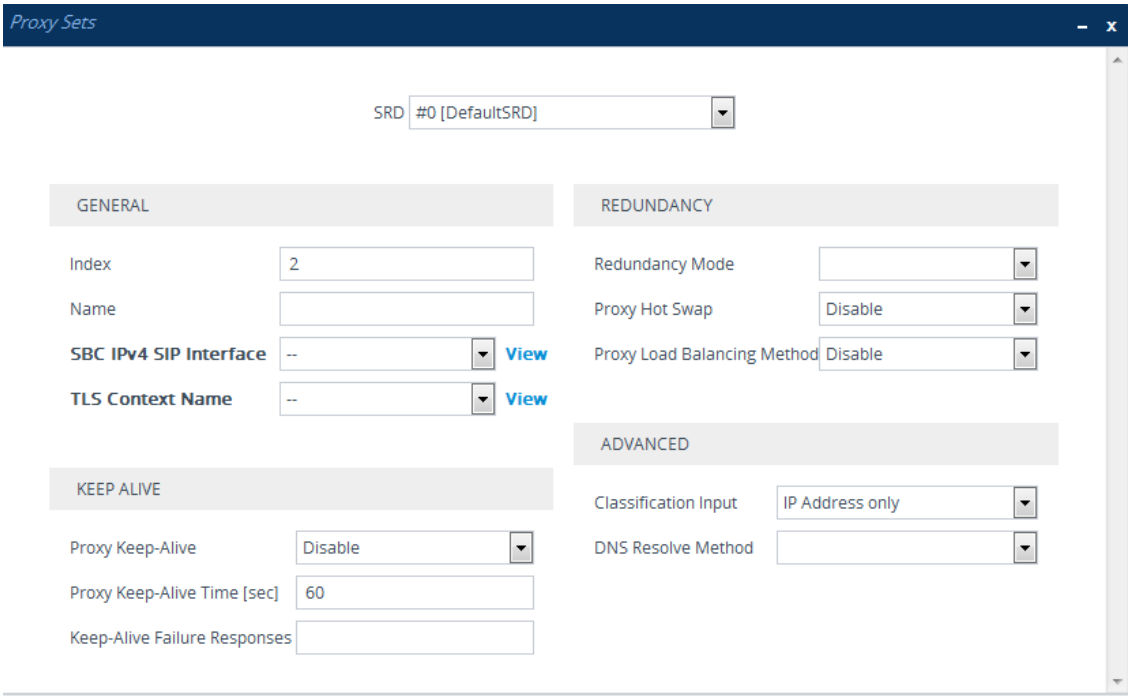
- It is recommended to classify incoming SIP dialogs to IP Groups based on Classification rules (see Configuring Classification Rules on page 703) instead of based on Proxy Sets.
- To view the connectivity status of Proxy Sets, see Viewing Proxy Set Status on page 968.

The Proxy Set is configured using two tables, one a "child" of the other:

- Proxy Sets table: Defines the attributes of the Proxy Set such as associated SIP Interface and redundancy features - ini file parameter [ProxySet] or CLI command, `configure voip > proxy-set`
- Proxy Set Address table ("child"): Defines the addresses of the Proxy Set - table ini file parameter [ProxyIP] or CLI command, `configure voip > proxy-ip > proxy-set-id`

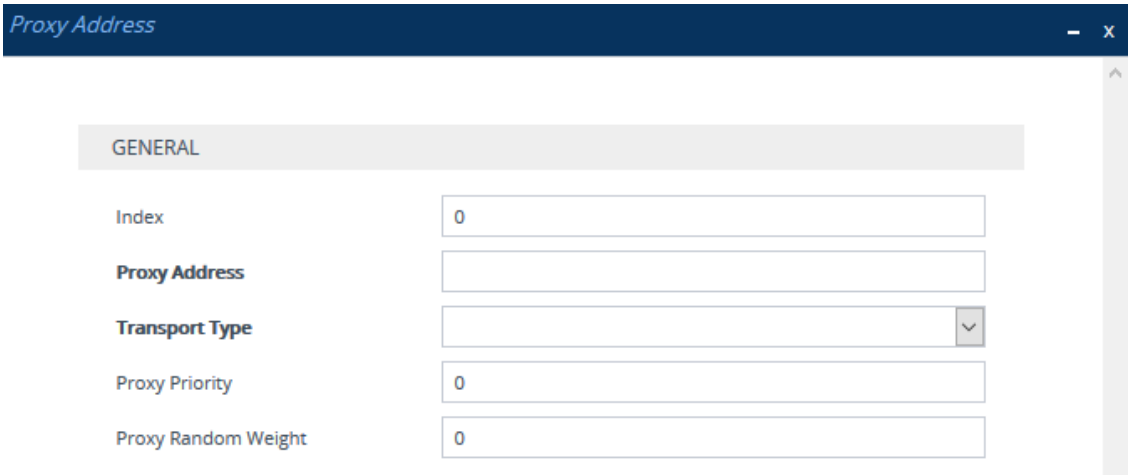
➢ To configure a Proxy Set:

1. Open the Proxy Sets table (**Setup** menu > **Signaling & Media** tab > **Core Entities** folder >**Proxy Sets**).
2. Click **New**; the following dialog box appears (screenshot has been cropped due to page size):



3. Configure a Proxy Set according to the parameters described in the table below.

4. Click **Apply**.
5. Select the index row of the Proxy Set that you added, and then click the **Proxy Address** link located below the table; the Proxy Address table opens.
6. Click **New**; the following dialog box appears:



7. Configure the address of the Proxy Set according to the parameters described in the table below.
8. Click **Apply**.

Table 18-7: Proxy Sets Table and Proxy Address Table Parameter Description

Parameter	Description
'SRD' `voip-network proxy-set > srd-id` [ProxySet_SRDName]	Assigns an SRD to the Proxy Set. **Note:** ■ The parameter is mandatory and must be configured first before you can configure the other parameters in the table. ■ To configure SRDs, see Configuring SRDs.
General	
'Index' `configure voip > voip-network proxy-set` [ProxySet_Index]	Defines an index number for the new table row. **Note:** Each row must be configured with a unique index.
'Name' `proxy-name` [ProxySet_ProxyName]	Defines a descriptive name, which is used when associating the row in other tables. The valid value is a string of up to 40 characters. **Note:**

Parameter	Description
	■ Each row must be configured with a unique name. ■ The parameter value cannot contain a forward slash (/).
'SBC IPv4 SIP Interface' `sbcipv4-sip-int-name` [ProxySet_ SBCIPv4SIPInterfaceName]	Assigns an IPv4-based SIP Interface for SBC calls to the Proxy Set. **Note:** ■ At least one SIP Interface must be assigned to the Proxy Set. ■ The parameter appears only if you have configured a network interface with an IPv4 address in the IP Interfaces table (see Configuring IP Network Interfaces). ■ To configure SIP Interfaces, see Configuring SIP Interfaces.
'SBC IPv6 SIP Interface' `sbcipv6-sip-int-name` [ProxySet_ SBCIPv6SIPInterfaceName]	Assigns an IPv6-based SIP Interface for SBC calls to the Proxy Set. **Note:** ■ At least one SIP Interface must be assigned to the Proxy Set. ■ The parameter appears only if you have configured a network interface with an IPv6 address in the IP Interfaces table.
'TLS Context Name' `tls-context-name` [ProxySet_ TLSContextName]	Assigns a TLS Context (TLS configuration) to the Proxy Set. By default, no TLS Context is assigned. If you assign a TLS Context, the TLS Context is used as follows: ■ Incoming calls: If the 'Transport Type' parameter (in this table) is set to **TLS** and the incoming call is successfully classified to an IP Group based on the Proxy Set, this TLS Context is used. If the 'Transport Type' parameter is set to **UDP** or classification to this Proxy Set fails, the TLS Context is not used. Instead, the device uses the TLS Context configured for the SIP Interface (see Configuring SIP Interfaces) used for the call; otherwise, the default TLS Context (ID 0) is used. ■ Outgoing calls: If the 'Transport Type' parameter is set to **TLS** and the outgoing call is sent to an IP Group that is associated with this Proxy Set, this TLS Context is used. Instead, the device uses the TLS Context configured for the SIP Interface used for the call; otherwise, the default TLS

Parameter	Description
	Context (ID 0) is used. If the 'Transport Type' parameter is set to **UDP**, the device uses UDP to communicate with the proxy and no TLS Context is used. To configure TLS Contexts, see Configuring TLS Certificates on page 158.
Keep Alive	
'Proxy Keep-Alive' `proxy-enable-keep-alive` [ProxySet_ EnableProxyKeepAlive]	Enables the device's Proxy Keep-Alive feature, which checks connectivity with all the proxy servers of the Proxy Set, by sending keep-alive messages. ■ [0] **Disable** (default). ■ [1] **Using OPTIONS** = Enables the Proxy Keep-Alive feature using SIP OPTIONS messages. The device sends an OPTIONS message every user-defined interval, configured by the 'Proxy Keep-Alive Time' parameter (in this table). If the device receives a SIP response code that is configured in the 'Keep-Alive Failure Responses' parameter (in this table), the device considers the proxy as offline. You can also configure if the device uses its IP address, the proxy's IP address, or the device's name in the OPTIONS message, using the [UseGatewayNameForOptions] parameter. ■ [2] **Using REGISTER** = Enables the Proxy Keep-Alive feature using SIP REGISTER messages. The device sends a REGISTER message every user-defined interval, configured by the SBCProxyRegistrationTime parameter. Any SIP response from the proxy - success (200 OK) or failure (4xx response) - is considered as if the proxy is "alive". If the proxy does not respond to INVITE messages sent by the device, the proxy is considered as down (offline). The device sends keep-alive REGISTER messages only to one proxy. Only if the proxy fails to respond to the keep-alive, does the device send the next keep-alive REGISTER message to another proxy. ■ [3] **Using OPTIONS on Active Server** = Enables the Proxy Keep-Alive feature using SIP OPTIONS messages (similar to the **Using OPTIONS** value), except that the proxy servers to which the keep-alive messages are sent depend on the settings of the Proxy Set's 'Redundancy Mode' parameter (see below): ✓ **Parking**: The device sends the keep-alive OPTIONS messages only to the currently active proxy server (to

Parameter	Description
	which it is connected and using). ✓ **Homing**: The device sends keep-alive OPTIONS messages to the currently active proxy server as well as to all proxy servers with higher priority (according to the 'Proxy Priority' parameter described below) than the active server. Once a higher priority server comes online, the device stops sending the keep-alive OPTIONS messages to the previously active server and connects to the higher priority server. The device now sends keep-alive messages to this newly active server and all other servers with higher priority. ✓ If the 'Redundancy Mode' parameter is not configured (empty) and the Proxy Set's 'Proxy Load Balancing Method' parameter (see below) is configured to any value other than **Disable**, the device sends the keep-alive OPTIONS messages to all proxy servers (same behavior as when you configure the 'Proxy Keep-Alive' parameter to **Using OPTIONS**). **Note:** ■ Proxy keep-alive using REGISTER messages (**Using REGISTER**) is applicable only to the Parking redundancy mode ('Redundancy Mode' parameter configured to **Parking**). ■ If you enable this Proxy Keep-Alive feature, the device can operate with multiple proxy servers (addresses) for redundancy and load balancing (see the 'Proxy Load Balancing Method' parameter). ■ For Survivability mode for User-type IP Groups, you must enable this Proxy Keep-Alive feature. ■ If you enable this Proxy Keep-Alive feature and the proxy uses the TCP/TLS transport type, you can enable CRLF Keep-Alive feature, using the [UsePingPongKeepAlive] parameter. ■ If you enable proxy keep-alive using SIP OPTIONS messages (**Using OPTIONS** or **Using OPTIONS on Active Server**), you can also enable the device to apply various settings (e.g., SIP message manipulations) of the IP Group that is associated with the Proxy Set , to these SIP messages. For more information, see the 'Proxy Keep-Alive using IP Group Settings' parameter in the IP Groups table.

Parameter	Description
	■ If you enable proxy keep-alive using SIP OPTIONS messages (**Using OPTIONS** or **Using OPTIONS on Active Server**), you can also configure how long the device waits (in seconds) before re-sending a keep-alive message once the device considers the proxy as offline (i.e., after all retransmissions, configured by the 'Failure Detection Retransmissions' have failed). This feature is configured by the [FailedOptionsRetryTime] parameter.
'Proxy Keep-Alive Time' `proxy-keep-alive-time` [ProxySet_ ProxyKeepAliveTime]	Defines the interval (in seconds) between keep-alive messages sent by the device when the Proxy Keep-Alive feature is enabled (see the 'Proxy Keep-Alive' parameter in this table). The valid range is 5 to 2,000,000. The default is 60. **Note:** The parameter is applicable only if the 'Proxy Keep-Alive' parameter is configured to **Using OPTIONS** or **Using OPTIONS on Active Server**.
'Keep-Alive Failure Responses' `keepalive-fail-resp` [ProxySet_ KeepAliveFailureResp]	Defines SIP response codes that if any is received in response to a keep-alive message using SIP OPTIONS ('Proxy Keep-Alive' configured to **Using OPTIONS** or **Using OPTIONS on Active Server**), the device considers the proxy as down. Up to three response codes can be configured, where each code is separated by a comma (e.g., 407,404). By default, no response code is defined. If no response code is configured, or if response codes received are not those configured, the proxy is considered online. **Note:** The SIP 200 response code is not supported for this feature.
'Success Detection Retries' `success-detect-retries` [ProxySet_ SuccessDetectionRetries]	Defines the minimum number of consecutive, successful keep-alive messages that the device sends to an offline proxy, before the device considers the proxy as being online. The interval between the sending of each consecutive successful keep-alive is configured by the 'Success Detection Interval' parameter (see below). For an example of using this parameter, see the 'Success Detection Interval' parameter. The valid range is 1 to 100. The default is 1. **Note:** The parameter is applicable only if the 'Proxy Keep-Alive' parameter is configured to **Using OPTIONS** or **Using OPTIONS on Active Server**.
'Success Detection Interval'	Defines the interval (in seconds) between each successful keep-alive retries (as configured by the 'Success Detection Retries'

Parameter	Description
`success-detect-int` [ProxySet_ SuccessDetectionInterval]	parameter) that the device performs for offline proxies. The valid range is 1 to 200. The default is 10. For example, assume that the 'Success Detection Retries' parameter is configured to 3 and the 'Success Detection Interval' parameter to 5 (seconds). When connectivity is lost with the proxy, the device sends keep-alive messages to the proxy. If the device receives a successful response from the proxy, it sends another (1st) keep-alive after 5 seconds, and if successful, sends another (2nd) keep-alive after 5 seconds, and if successful, sends another (3rd) keep-alive after 5 seconds, and if successful, considers connectivity with the proxy as being restored. **Note:** The parameter is applicable only if the 'Proxy Keep-Alive' parameter is configured to **Using OPTIONS** or **Using OPTIONS on Active Server**.
'Failure Detection Retransmissions' `fail-detect-rtx` [ProxySet_ FailureDetectionRetransmissions]	Defines the maximum number of UDP retransmissions that the device sends to an offline proxy before the device considers the proxy as offline. The valid range is -1 to 255. The default is -1, which means that the settings of the global parameter [SIPMaxRtx] is applied. **Note:** ■ The parameter is applicable only if the 'Proxy Keep-Alive' parameter is configured to **Using OPTIONS** or **Using OPTIONS on Active Server**. ■ If the receives an ICMP error response (which indicates Host Unreachable or Network Unreachable) as opposed to a timeout, it may be desirable to abandon additional retries in favor of trying the next IP address (proxy) in the Proxy Set (typically required when Proxy Hot Swap is enabled). To enable this, configure the [AbortRetriesOnICMPError] parameter to 1.
Redundancy	
'Redundancy Mode' `proxy-redundancy-mode` [ProxySet_ ProxyRedundancyMode]	Enables the proxy redundancy mode. ■ [-1] = Not configured (Default). Proxy redundancy method is according to the settings of the global parameter [ProxyRedundancyMode]. ■ [0] **Parking** = If the device operates with a proxy server that has the highest priority and the proxy goes offline, the

Parameter	Description
	device attempts to connect and operate with a different proxy that has the highest priority of all currently online proxies. However, once the device starts operating with this new proxy, it remains operating with it even if a previously offline proxy that has higher priority becomes online again. ■ [1] **Homing** = The device always attempts to operate with the proxy that has the highest priority of all currently online proxies. For example, if the device is currently operating with proxy server 200.10.1.1 that has priority 4, and then a previously offline proxy 200.10.1.2 that has priority 0 (i.e., a higher priority) becomes online again, the device attempts to connect and operate with proxy 200.10.1.2. **Note:** ■ For proxy redundancy, you also need to enable the proxy keep-alive feature (see the 'Proxy Keep-Alive' parameter, above). The **Homing** redundancy mode is applicable only to proxy keep-alive using SIP OPTIONS (i.e., 'Proxy Keep-Alive' parameter is configured to **Using OPTIONS** or **Using OPTIONS on Active Server**). The **Parking** redundancy mode is applicable to all proxy keep-alive methods. ■ From Version 7.20A.204, if you configure the parameter to **Parking** and the proxy keep-alive is done using REGISTER messages, when the proxy goes offline, the device arbitrarily chooses the next proxy to operate with. ■ To configure proxy priority, see the 'Proxy Priority' parameter in the Proxy Address table (below).
'Proxy Hot Swap' `is-proxy-hot-swap` [ProxySet_ IsProxyHotSwap]	Enables the Proxy Hot-Swap feature, whereby if the device sends a SIP message (INVITE or REGISTER) to the proxy and the message fails, the device re-sends the same message to a redundant proxy configured for the Proxy Set. The redundant proxy is determined by your Proxy Set configuration (i.e., redundancy mode and load balancing). ■ [0] **Disable** = (Default) Disables the Proxy Hot-Swap feature. If the device sends a SIP message (INVITE or REGISTER) to the proxy and the proxy rejects it or no response is received from the proxy for a user-defined number of re-transmissions, configured by the [SIPMaxRtx] parameter, the device does not attempt to connect to any other proxy in the Proxy Set and the SIP message fails.

Parameter	Description
	However, if you have configured an SBC Alternative Routing Reasons Set for the IP Group (see Configuring SIP Response Codes for Alternative Routing Reasons), the device tries all proxies in the Proxy Set. If it successfully connects to one of the redundant proxies, it re-sends the message to this proxy. This functionality doesn't apply to REGISTER requests initiated by the device (e.g., for Accounts). ■ [1] **Enable** = If the device sends a SIP message (INVITE or REGISTER) to the proxy with which it is currently operating and any of the following occurs, the device re-sends the message to a redundant online proxy: ✓ No response is received from the proxy each time the device re-sends it. The number of retransmissions is configured by the [HotSwapRtx] parameter. In such a scenario, the device issues itself the SIP response code 408 (Request Timeout). ✓ The proxy rejects the message with a SIP response code that you have also configured for the Alternative Reasons Set that is assigned to the IP Group ('SBC Alternative Routing Reasons Set' parameter) associated with the Proxy Set (see Configuring SIP Response Codes for Alternative Routing Reasons). **Note:** You can employ alternative routing with this option. If no response is received from any of the redundant (online) proxies or the proxies reject the message with a SIP response code that you have configured for the Alternative Reasons Set that is assigned to the IP Group ('SBC Alternative Routing Reasons Set' parameter) associated with the Proxy Set, the device searches the IP-to-IP Routing table for an alternative routing rule and if found, sends the message to the rule's destination. For more information on the Proxy Hot Swap feature and alternative routing based on SIP response codes, see Configuring SIP Response Codes for Alternative Routing Reasons on page 742.
'Proxy Load Balancing Method' `proxy-load-balancing-method` [ProxySet_	Enables load balancing between proxy servers of the Proxy Set. ■ [0] **Disable** = (Default) Disables proxy load balancing. ■ [1] **Round Robin** = The device sends outgoing SIP messages to the online proxy servers of the Proxy Set in a round-robin fashion. The order of the round-robin is determined by the

Parameter	Description
ProxyLoadBalancingMethod]	listed order of the IP addresses in the Proxy Address table and their priority. You can configure priority of each IP address using the 'Proxy Priority' parameter (see below). For DNS-resolved IP addresses for proxy servers configured with an FQDN (including NAPTR and SRV, if configured), the priority is received from the DNS.The IP address list is refreshed every user-defined interval (see the ProxyIPListRefreshTime parameter). If a change in the order of the IP address entries in the list occurs, all load statistics are erased and balancing starts over again. ■ [2] **Random Weights** = The outgoing requests are not distributed equally among the proxy servers. The distribution is determined by the weight of the proxy servers. You can configure the weight per proxy server, using the 'Proxy Random Weight' parameter in the Proxy Address table (see below). For proxy servers configured with an FQDN, the weight of each DNS-resolved IP address is received from the DNS server (using SRV records). However, if you have configured the weight for the FQDN in the 'Proxy Random Weight' parameter, this parameter's value overrides the weight from the DNS server. The device sends the requests in such a fashion that each proxy receives a percentage of the requests according to its' weight.
'Min. Active Servers for Load Balancing' `min-active-serv-lb` [ProxySet_ MinActiveServersLB]	Defines the minimum number of proxies in the Proxy Set that must be online for the device to consider the Proxy Set as online, when proxy load balancing is used. The valid value is 1 to 15. The default is 1. **Note:** The parameter is applicable only if proxy load balancing is enabled (see the 'Proxy Load Balancing Method' parameter, above).
Advanced	
'Classification Input' `classification-input` [ProxySet_ ClassificationInput]	Defines how the device classifies incoming IP calls to the Proxy Set. ■ [0] **IP Address only** = (Default) Classifies calls to the Proxy Set according to IP address only. ■ [1] **IP Address, Port & Transport Type** = Classifies calls to the Proxy Set according to IP address, port, and transport type.

Parameter	Description
	Note: ■ The parameter is applicable only if the IP Groups table's parameter 'Classify by Proxy Set' is configured to **Enable** (see Configuring IP Groups). ■ If multiple Proxy Sets are configured with the same IP address and associated with the same SIP Interface, the device may classify the SIP dialog (based on Proxy Set) to an incorrect IP Group. In such a scenario, the device uses the Proxy Set with the lowest Index number (e.g., Proxy Set ID #1 over Proxy Set ID #4). A Syslog warning message is generated in such scenarios. Therefore, it is recommended to configure each Proxy Set with a unique IP address. If multiple Proxy Sets are configured with the same IP address but associated with different SIP Interfaces, then classification based on Proxy Set can be correctly achieved. If multiple Proxy Sets are configured with the same IP address and SIP Interface, but with different ports (e.g., 10.1.1.1:5060 and 10.1.1.1:5070) and the parameter is configured to **IP Address, Port & Transport Type**, classification to the correct IP Group is achieved. Therefore, when classification is by Proxy Set, pay attention to the configured IP addresses and this parameter. When multiple Proxy Sets are configured with the same IP address, the device selects the matching Proxy Set in the following order: ✓ Selects the Proxy Set whose IP address, port, and transport type match the source of the incoming SIP request (regardless of the settings of this parameter). ✓ If no match is found for above, it selects the Proxy Set whose IP address and transport type match the source of the incoming SIP request (if the parameter is configured to **IP Address Only**). ✓ If no match is found for above, it selects the Proxy Set whose IP address match the source of the incoming SIP request (if the parameter is configured to **IP Address Only**). For example: Index — Classification Input — Proxy Address (IP:Port;Transport Type)

Parameter	Description
	1 **IP Address, Port & Transport Type** 10.10.10.10:5060;UDP 2 **IP Address only** 10.10.10.10:5060;UDP 3 **IP Address only** 10.10.10.10:5070;UDP 4 **IP Address only** 10.10.10.10:5060;TCP ✓ **Incoming SIP request from 10.10.10.10:5060;UDP:** Best match is #1 and #2 (same priority); second best match is #3 (due to transport type); third best match is #4. ✓ **Incoming SIP request from 10.10.10.10:5080;TLS:** Best match is #2, #3 and #4 (same priority). ✓ **Incoming SIP request from 10.10.10.10:5070;TCP:** Best match is #4 (due to transport type); second best match is #2 and #3 (same priority).
'DNS Resolve Method' `dns-resolve-method` [ProxySet_ DNSResolveMethod]	Defines the DNS query record type for resolving the proxy server's host name (FQDN) into an IP address(es). ■ [-1] = Not configured. DNS resolution method is according to the settings of the global parameter [ProxyDNSQueryType]. ■ [0] **A-Record** = (Default) DNS A-record query is used to resolve DNS to IP addresses. ■ [1] **SRV** = If the proxy address is configured with a domain name without a port (e.g., domain.com), an SRV query is done. The SRV query returns the host names (and their weights). The device then performs DNS A-record queries per host name (according to the received weights). If the configured proxy address contains a domain name with a port (e.g., domain.com:5080), the device performs a regular DNS A-record query. ■ [2] **NAPTR** = NAPTR query is done. If successful, an SRV query is sent according to the information received in the NAPTR response. If the NAPTR query fails, an SRV query is done according to the configured transport type. If the configured proxy address contains a domain name with a port (e.g., domain.com:5080), the device performs a regular DNS A-record query. If the transport type is configured for the proxy address, a NAPTR query is not performed. ■ [3] **Microsoft Skype for Business** = An SRV query is done as

Parameter	Description
	required by Microsoft when the device is deployed in a Microsoft Skype for Business environment. The device sends a special SRV query to the DNS server according to the transport protocol configured in the 'Transport Type' parameter (described later in this section): ✓ **TLS** ✓ **TCP:** "_sipinternal._tcp.<domain>" and "_sip_ tcp.<domain>". ✓ **Undefined:** "_sipinternaltls_tcp.<domain>", "_ sipinternal_tcp.<domain>", "_sip_tls.<domain>" and "_ sip_tcp.<domain>". The SRV query returns the host names (and their weights). The device then performs DNS A-record queries per host name (according to the received weights) to resolve into IP addresses. **Note:** The device caches the DNS-resolved IP addresses of the last successful DNS query. For more information, see the description of the [ProxyIPListRefreshTime] parameter.
'Accept DHCP Proxy List' `accept-dhcp-proxy-list` [ProxySet_ AcceptDHCPProxyList]	Enables the device (acting as a DHCP client) to obtain the Proxy Set's address(es) from a DHCP server. ■ [0] **Disable** (default) ■ [1] **Enable** = The device sends a DHCP request with Option 120 (SIP server address) to a DHCP server. This occurs upon a DHCP refresh (lease renewal). When the device receives the list of IP addresses (or DNS) from the server, it adds them to the Proxy Set (replacing any existing IP addresses or DNS). **Note:** When enabled, the device uses UDP and port 5060.
Proxy Address Table	
'Index' `proxy-ip-index` [ProxyIp_ProxyIpIndex]	Defines an index number for the new table row. **Note:** Each row must be configured with a unique index.
'Proxy Address' `proxy-address` [ProxyIp_IpAddress]	Defines the address of the proxy server (Proxy Set). The address can be defined as an IP address in dotted-decimal notation (e.g., 201.10.8.1) or FQDN. You can also specify the port using the following format:

Parameter	Description
	■ IPv4 address: <IP address>:<port> (e.g., 201.10.8.1:5060) ■ IPv6 address: <[IPV6 address]>:<port> (e.g., [2000::1:200:200:86:14]:5060) **Note:** ■ When configured with an FQDN, you can configure the periodic interval at which the device performs DNS queries to resolve the FQDN into IP addresses. For more information, see the [ProxyIPListRefreshTime] parameter. ■ When configured with an FQDN, you can configure the method (e.g., A-record) for resolving the domain name into an IP address, using the 'DNS Resolve Method' parameter in this table (see above). ■ You can configure the device to use the port indicated in the Request-URI of the incoming message, instead of the port configured for the parameter. To enable this, use the [IPGroup_SBCRouteUsingRequestURIPort] parameter for the IP Group that is associated with the Proxy Set (Configuring IP Groups). ■ If you are configuring the Proxy Sets with IP addresses, it is highly recommended to configure each Proxy Set with a unique IP address. Configuring multiple Proxy Sets with the same IP address can cause problems classifying incoming SIP requests to source IP Groups based on Proxy Set. If you have configured multiple Proxy Sets with the same IP address, the device uses the Proxy Set with lowest Index number. For example, if you have configured Proxy Set ID #1 and Proxy Set ID #4 with the same IP address, the device uses Proxy Set ID #1 to classify the incoming SIP request to an IP Group. However, configuring multiple Proxy Sets with the same IP address, but with different SIP Interfaces is acceptable for classifying incoming SIP requests to source IP Groups based on Proxy Set. For more information on determining the Proxy Set, see the 'Classification Input' parameter (above) parameter .
'Transport Type' `transport-type` [ProxyIp_	Defines the transport type for communicating with the proxy. ■ [-1] = (Default) Not configured. The transport type is according to the settings of the global parameter

Parameter	Description
TransportType]	[SIPTransportType]. ■ [0] **UDP** ■ [1] **TCP** ■ [2] **TLS**
'Proxy Priority' priority [ProxyIp_Priority]	Defines the priority of the proxy. When a proxy server goes offline, the device attempts to connect to an online proxy server that has the highest priority. The valid value is 0 to 65535, where 0 is the highest priority and 65535 the lowest. The default is 0. **Note:** ■ You must configure both priority and weight (or none of them). In other words, if you configure this parameter, you must also configure the 'Proxy Random Weight' parameter. If you don't configure this parameter, you must also not configure the 'Proxy Random Weight' parameter. ■ If weight and priority are not configured for any of the proxy servers of the Proxy Set, the order in which the addresses (IP addresses and FQDNs) are listed in the table determine their priority (i.e., top-listed address has the highest priority). ■ For FQDNs, weight and priority of DNS-resolved IP addresses are determined by the DNS. However, this parameter's value overrides the priority received from the DNS. ■ If you have configured at least one of the proxy servers of the Proxy Set with weight and priority, the device prioritizes all the configured proxy servers according to weight and priority. In this case, proxy servers that are not configured with priority (i.e., 0) are considered as proxy servers with the highest priority. ■ The parameter is applicable to load balancing (see the 'Proxy Load Balancing Method' parameter), and homing and parking redundancy (see the 'Redundancy Mode' parameter).
'Proxy Random Weight' weight [ProxyIp_Weight]	Defines the weight of the proxy. The valid value is 0 to 65535, where 0 is the highest weight and 65535 the lowest. The default is 0.

Parameter	Description
	Note: ■ The parameter is applicable only if you configure the 'Proxy Load Balancing Method' parameter to **Random Weights**. For more information on weights, see this parameter. ■ You must configure both priority and weight (or none of them). In other words, if you configure this parameter, you must also configure the 'Proxy Priority' parameter. If you don't configure this parameter, you must also not configure the 'Proxy Priority' parameter. ■ For proxy servers configured with FQDNs, this parameter's value overrides the weight received for DNS-resolved IP addresses from the DNS server.

Building and Viewing SIP Entities in Topology View

The Topology view lets you easily build and view your main SIP entities, including IP Groups, SIP Interfaces, and Media Realms. The Topology view graphically displays these entities and the associations between them, giving you a better understanding of your SIP topology and configuration. The Topology view also lets you configure additional SIP settings that are important to your deployment such as routing and manipulation. You can use the Topology view as an alternative to configuring the entities in their respective Web pages or you can use it in combination.

To access the Topology view, do one of the following:

- Click the Topology View home icon (**Setup** menu > **Signaling & Media** tab > **Topology View**).
- Click the logo, which is located in the top-left corner of the Web interface.

The main areas of the Topology view is shown below and described in the subsequent table.

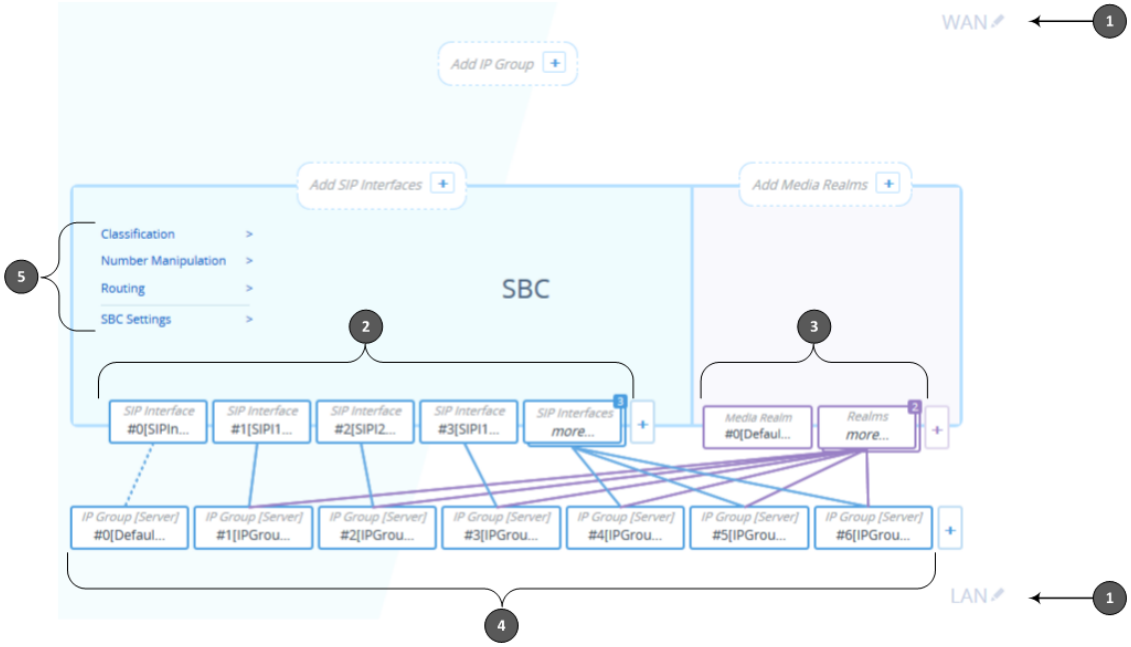
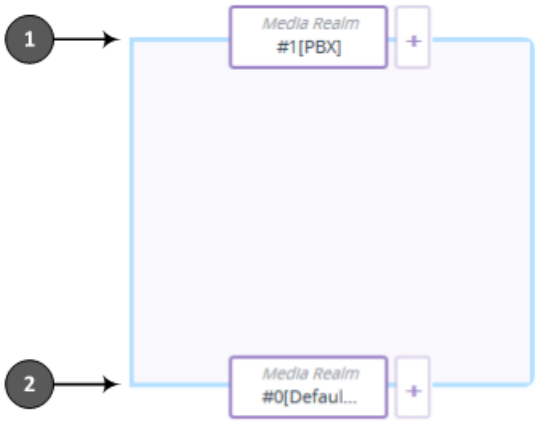
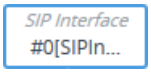
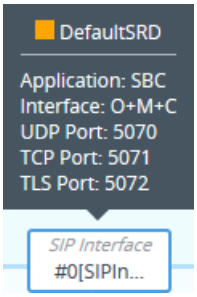
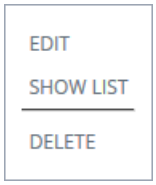
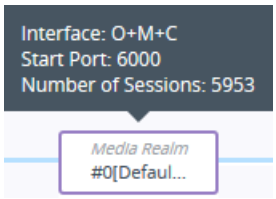
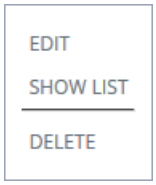
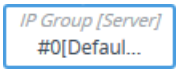
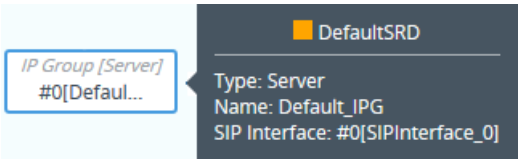
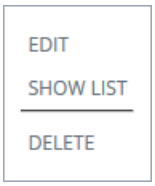


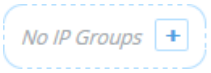
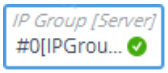
Table 18-8: Description of Topology View

Item #	Description
1	Demarcation area of the topology. By default, the Topology view displays the following names to represent the different demarcations of your voice configuration: ■ "WAN": Indicates the external network side ■ "LAN": Indicates the internal network (e.g., inside the company) To modify a demarcation name, do the following: 1. Click the demarcation name; the name becomes editable in a text box, as shown in the example below: 2. Type a name as desired, and then click anywhere outside of the text box to apply the name. You can use demarcation to visually separate your voice network to provide a clearer understanding of your topology. This is especially useful for IP Groups, SIP Interfaces, and Media Realms, where you can display them on the top or bottom border of the Topology view (as shown in the figure below for callouts #1 and #2, respectively). For example, on the top border you can position all entities relating to WAN, and on the bottom border all entities relating to LAN.

Item #	Description
	 By default, configuration entities are displayed on the bottom border. To define the position, use the 'Topology Location' parameter when configuring the entity, where **Down** is the bottom border and **Up** the top border: 
2	Configured SIP Interfaces. Each SIP Interface is displayed using the following "SIP Interface"-titled icon, which includes the name and row index number:  If you hover your mouse over the icon, a pop-up appears displaying the following basic information (example):  If you click the icon, a drop-down menu appears listing the following commands:  - **Edit:** Opens a dialog box in the SIP Interfaces table to modify the SIP Interface. - **Show List:** Opens the SIP Interfaces table. - **Delete:** Opens the SIP Interfaces table where you are prompted to confirm

Item #	Description
	deletion of the SIP Interface. To add a SIP Interface, do the following: 1. Click the **Add SIP Interface** plus icon. The icon appears next to existing SIP Interfaces, or as when no SIP Interfaces exist on a topology border, or as when there are no SIP Interfaces at all. The SIP Interfaces table opens with a new dialog box for adding a SIP Interface to the next available index row. 2. Configure the SIP Interface as desired, and then click **Apply**; the SIP Interfaces table closes and you are returned to the Topology View, displaying the new SIP Interface. For more information on configuring SIP Interfaces, see Configuring SIP Interfaces.
3	Configured Media Realms. Each Media Realm is displayed using the following "Media Realm"-titled icon, which includes the name and row index number:  If you hover your mouse over the icon, a pop-up appears displaying the following basic information (example):  If you click the icon, a drop-down menu appears listing the following commands:  ■ **Edit:** Opens a dialog box in the Media Realms table to modify the Media Realm. ■ **Show List:** Opens the Media Realms table. ■ **Delete:** Opens the Media Realms table where you are prompted to confirm deletion of the Media Realm. To add a Media Realm, do the following:

Item #	Description
	1. Click the **Add Media Realm** plus icon. The icon appears next to existing Media Realms, or as when no Media Realms exist on a topology border, or as when there are no Media Realms at all. The Media Realms table opens with a new dialog box for adding a Media Realm to the next available index row. 2. Configure the Media Realm as desired, and then click **Apply**; the Media Realms table closes and you are returned to the Topology View, displaying the new Media Realm. For more information on configuring Media Realms, see Configuring Media Realms.
4	Configured IP Groups. Each IP Group is displayed using the following "IP Group [Server]" or "IP Group [User]" titled icon (depending on whether it's a Server- or User-type IP Group respectively), which includes the name and row index number (example of a Server-type):  If you hover your mouse over the icon, a pop-up appears displaying the following basic information (example):  If you click the icon, a drop-down menu appears listing the following commands:  ■ **Edit:** Opens a dialog box in the IP Groups table to modify the IP Group. ■ **Show List:** Opens the IP Groups table. ■ **Delete:** Opens the IP Groups table where you are prompted to confirm deletion of the IP Group. To add an IP Group, do the following: 1. Click the **Add IP Group** plus icon. The icon appears next to existing IP

Item #	Description
	Groups, or as  when no IP Groups exist on a topology border, or as  when there are no IP Groups at all. The IP Groups table opens with a new dialog box for adding a IP Group to the next available index row. 2. Configure the IP Group as desired, and then click **Apply**; the IP Groups table closes and you are returned to the Topology View, displaying the new IP Group. For more information on configuring IP Groups, see Configuring IP Groups. IP Group icons also display connectivity status with Server-type IP Groups: ■  (Green with check mark): Keep-alive is successful and connectivity exists with IP Group. ■  (Red with "x"): Keep-alive has failed and there is a loss of connectivity with the IP Group. The line type connecting between an IP Group and a SIP Interface indicates whether a routing rule has been configured for the IP Group. A solid line indicates that you have configured a routing rule for the IP Group; a dashed line indicates that you have yet to configure a routing rule. **Note:** ■ You can also view connectivity status in the IP Groups table. ■ To support the connectivity status feature, you must enable the keep-alive mechanism for the Proxy Set that is associated with the IP Group (see Configuring Proxy Sets). ■ The green-color state also applies to scenarios where the device rejects calls with the IP Group due to low QoE (e.g., low MOS), despite connectivity.
5	Links to Web pages relating to commonly required SBC configuration: ■ Classification: Opens the Classification table where you can configure Classification rules (see Configuring Classification Rules). ■ Number Manipulation: Opens the Outbound Manipulations table where you can configure manipulation rules on SIP Request-URI user parts (source or destination) or calling names in outbound SIP dialog requests (see Configuring IP-to-IP Outbound Manipulations). ■ Routing: Opens the IP-to-IP Routing table where you can configure IP-to-IP

Item #	Description
	routing rules (see Configuring SBC IP-to-IP Routing Rules). ■ SBC Settings: Opens the SBC General Settings page where you can configure miscellaneous settings.

19 Coders and Profiles

This section describes configuration of coders and SIP profiles.

Configuring Coder Groups

The Coder Groups table lets you configure up to 21 *Coder Groups*. The Coder Group determines the audio (voice) coders used for calls. Each Coder Group can include up to 10 coders, where the packetization time (ptime), bit rate, payload type, and silence suppression can be configured per coder. The first coder in the Coder Group has the highest priority and is used by the device whenever possible. If the remote side cannot use the first coder, the device attempts to use the next coder in the Coder Group, and so on.

The Coder Groups table provides a pre-defined Coder Group (index 0) that is configured with the G.711 A-law coder. If no other Coder Groups are configured, the default Coder Group (which you can modify) is used for all calls. Alternatively, if you want to use specific coders or coder settings (e.g., packetization time) for different calls (entities), you need to configure a Coder Group for each entity and then assign each Coder Group to an IP Profile (see Configuring IP Profiles) associated with the entity (IP Group). If an IP Group is not associated with a Coder Group, the default Coder Group is used.

You can also use Coder Groups for audio coder transcoding of SBC calls. If two SIP entities need to communicate, but one does not support a coder required by the other, the device can add the required coder to the SDP offer. The added coder is referred to as an extension coder. For more information on extension coders, see Coder Transcoding.

To apply a Coder Group for transcoding to a SIP entity:

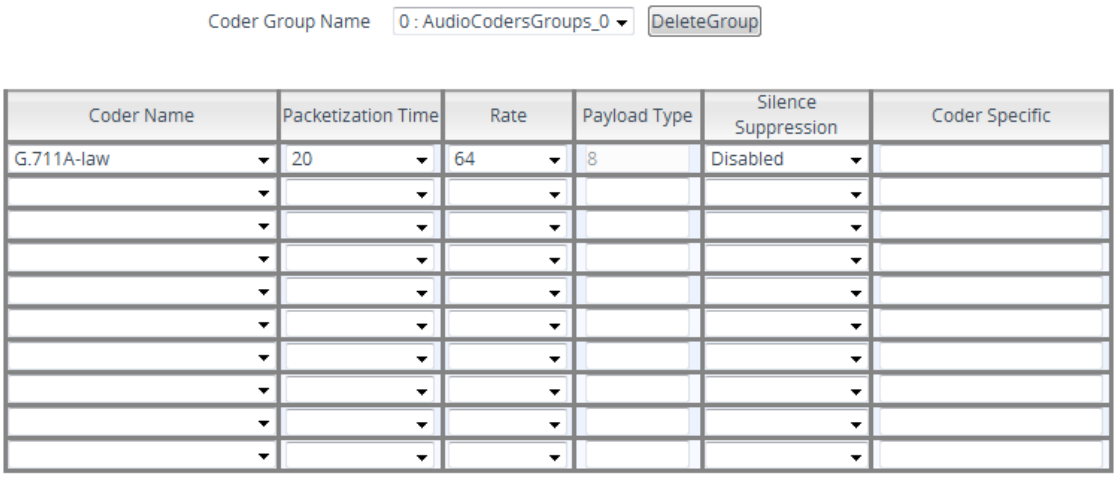
1. Configure a Coder Group in the Coder Groups table (see description below).
2. In the IP Profile associated with the SIP entity (see Configuring IP Profiles):
   - Assign the Coder Group (using the IpProfile_SBCExtensionCodersGroupName parameter).
   - Enable the use of the Coder Group for transcoding (by configuring the IpProfile_ SBCAllowedCodersMode parameter to Restriction or Restriction and Preference).

- For supported audio coders, see Supported Audio Coders.
- Some coders are license-based and are available only if included in the License Key installed on your device. For more information, contact the sales representative of your purchased device.
- Only the packetization time of the first coder listed in the Coder Group is declared in INVITE/200 OK SDP even if multiple coders are configured. The device always uses the packetization time requested by the remote side for sending RTP packets. If not specified, the packetization time is assigned the default value.
- The value of some fields is hard-coded according to common standards (e.g., payload type of G.711 U-law is always 0).
- The G.722 coder provides Packet Loss Concealment (PLC) capabilities, ensuring higher voice quality.
- Opus coder:
  - ✓ For SBC calls: If one leg uses a narrowband coder (e.g., G.711) and the other leg uses the Opus coder, the device maintains the narrowband coder flavor by using the narrowband Opus coder. Alternatively, if one leg uses a wideband coder (e.g., G.722) and the other leg uses the Opus coder, the device maintains the wideband coder flavor by using the wideband Opus coder.
  - ✓ Gateway calls always use the narrowband Opus coder.
- For more information on V.152 and implementation of T.38 and VBD coders, see Supporting V.152 Implementation.
- The G.729 coder refers to G.729A if silence suppression is disabled, or G.729AB if silence suppression is enabled.

The following procedure describes how to configure the Coder Groups table through the Web interface. You can also configure it through ini file [AudioCodersGroups] and [AudioCoders], or CLI (`configure voip > coders-and-profiles audio-coders-groups`).

➢ To configure a Coder Group:

1. Open the Coder Groups table (**Setup** menu > **Signaling & Media** tab > **Coders & Profiles** folder > **Coder Groups**).



2. From the 'Coder Group Name' drop-down list, select the desired Coder Group index number and name.
3. Configure the Coder Group according to the parameters described in the table below.
4. Click **Apply**, and then reset the device with a save-to-flash for your settings to take effect.

You can delete a Coder Group, as described in the following procedure.

➢ To delete a Coder Group:

1. From the 'Coder Group Name' drop-down list, select the Coder Group that you want to delete.
2. Click **Delete Group**.

Table 19-1: Coder Groups Table Parameter Descriptions

Parameter	Description
'Coder Group Name' [AudioCodersGroups_Index] [AudioCodersGroups_Name]	Defines the name and index for the Coder Group. **Note:** The Coder Group index/name cannot be configured.
[AudioCoders_ AudioCodersIndex]	Index row of the coder per Coder Group **Note:** The parameter is applicable only to the ini file.
'Coder Name' `name` [AudioCoders_Name]	Defines the coder type. For coder names, see Supported Audio Coders. **Note:** Each coder type (e.g., G.729) can be configured only once in the table.
'Packetization Time' `p-time` [AudioCoders_pTime]	Defines the packetization time (in msec) for the coder. The packetization time determines how many coder payloads are combined into a single RTP packet. For ptime, see Supported Audio Coders.
'Rate' `rate` [AudioCoders_rate]	Defines the bit rate (in kbps) for the coder. For rates, see Supported Audio Coders.
'Payload Type' `payload-type` [AudioCoders_PayloadType]	Defines the payload type if the payload type (i.e., format of the RTP payload) for the coder is dynamic. For payload types, see Supported Audio Coders.
'Silence Suppression' `silence-suppression` [AudioCoders_Sce]	Enables silence suppression for the coder. ■ [0] **Disable** (Default) ■ [1] **Enable**

Parameter	Description
	■ [2] **Enable w/o Adaptation** = Enables silence suppression using AudioCodes proprietary noise adaptation mechanism. This is applicable only when any of the following coders are used: ✓ G.711: The device sends only one SID packet during periods of silence. **Note:** ■ If you disable silence suppression for G.729, the device includes 'annexb=no' in the SDP of the relevant SIP messages. If you enable silence suppression, 'annexb=yes' is included.
'Coder Specific' `coder-specific` [AudioCoders_CoderSpecific]	Defines additional settings specific to the coder. Currently, the parameter is applicable only to the AMR coder and is used to configure the payload format type. ■ [0] **0** = Bandwidth Efficient ■ [1] **1** = Octet Aligned (default) Note: The AMR payload type can be configured globally using the AmrOctetAlignedEnable parameter. However, the Coder Group configuration overrides the global parameter.

Supported Audio Coders

The table below lists the coders supported by the device.

Table 19-2: Supported Audio Coders

Coder Name	Packetization Time (msec) [1] 10, [2] 20, [3] 30, [4] 40, [5] 50, [6] 60, [8] 80, [9] 90, [10] 100, [12] 120	Rate (kbps)	Payload Type	Silence Suppression
G.723.1 `g723-1` [0]	30 (default), 60, 90, 120, 150	■ [7] 5.3 (default) ■ [11] 6.3	4	■ [0] Disable (default) ■ [1] Enable

Coder Name	Packetization Time (msec) [1] 10, [2] 20, [3] 30, [4] 40, [5] 50, [6] 60, [8] 80, [9] 90, [10] 100, [12] 120	Rate (kbps)	Payload Type	Silence Suppression
G.711 A-law g711-alaw [1]	10, 20 (default), 30, 40, 50, 60, 80, 100, 120	[90] 64	8	■ [0] Disable (default) ■ [1] Enable
G.711 U-law g711-ulaw [2]	10, 20 (default), 30, 40, 50, 60, 80, 100, 120	[90] 64	0	■ [0] Disable (default) ■ [1] Enable
G.729 g729 [3]	10, 20 (default), 30, 40, 50, 60, 80, 100	[19] 8	18	■ [0] Disable (default) ■ [1] Enable ■ [2] Enable w/o Adaptations
T.38 t-38 [4]	N/A	N/A	N/A	N/A (Disabled)
G.726 g726 [5]	10, 20 (default), 30, 40, 50, 60, 80	■ [43] 16 ■ [57] 24 ■ [64] 32 (default) ■ [70] 40	Dynamic (default 2)	■ [0] Disable (default) ■ [1] Enable
AMR amr [14]	20 (default)	■ [4] 4.75 ■ [6] 5.15 ■ [9] 5.90 ■ [14] 6.70	Dynamic	■ [0] Disable ■ [1] Enable

Coder Name	Packetization Time (msec) [1] 10, [2] 20, [3] 30, [4] 40, [5] 50, [6] 60, [8] 80, [9] 90, [10] 100, [12] 120	Rate (kbps)	Payload Type	Silence Suppression
		■ [16] 7.40 ■ [18] 7.95 ■ [27] 10.2 ■ [30] 12.2 (default)		
AMR-WB amr-wb [15]	20 (default)	■ [13] 6.6 ■ [21] 8.85 ■ [32] 12.65 ■ [37] 14.25 ■ [41] 15.85 ■ [48] 18.25 ■ [49] 19.85 ■ [53] 23.05 ■ [55] 23.8 (default)	Dynamic	■ [0] Disable ■ [1] Enable
iLBC ilbc [19]	20 (default), 40, 60, 80, 100, 120	[39] 15 (default)	Dynamic (default 65)	■ [0] Disable ■ [1] Enable
	30 (default), 60, 90, 120	[34] 13		
G.722 g722 [20]	10 (default), 20, 30, 40, 50, 60, 80, 100, 120	■ [90] 64 (default) ■ [74] 48 ■ [80] 56	■ 9 (applicable only to rate 64 kbps) ■ 66 (default and applicable only to rate	N/A (Disabled)

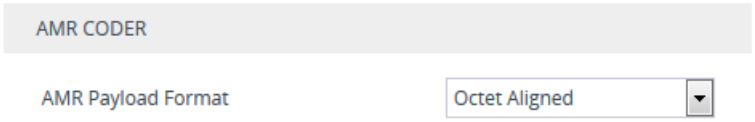
Coder Name	Packetization Time (msec) [1] 10, [2] 20, [3] 30, [4] 40, [5] 50, [6] 60, [8] 80, [9] 90, [10] 100, [12] 120	Rate (kbps)	Payload Type	Silence Suppression
			48 kbps) - payload can be changed ■ 67 (default and applicable only to rate 56 kbps) - payload can be changed	
G.711A-law_VBD g711a-law-vbd [23]	10, 20 (default), 30, 40, 50, 60, 80, 100, 120	[90] 64	8 or Dynamic (default 118)	N/A (Disabled)
G.711U-law_VBD g711u-law-vbd [24]	10, 20 (default), 30, 40, 50, 60, 80, 100, 120	[90] 64	0 or Dynamic (default 110)	N/A (Disabled)
SILK-NB silk-nb [35]	20 (default), 40, 60, 80, and 100	[19] 8	Dynamic (default 76)	N/A
SILK-WB silk-wb [36]	20 (default), 40, 60, 80, and 100	[43] 16	Dynamic (default 77	N/A
Opus opus [40]	20 (default), 40, 60, 80, 120	N/A	Dynamic (default 111)	N/A

Configuring Various Codec Attributes

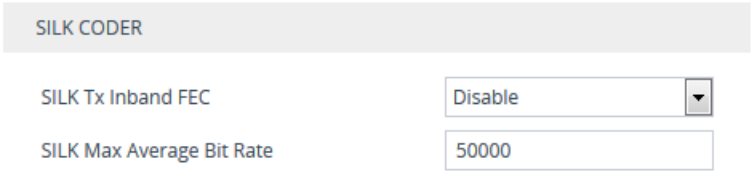
The following procedure describes how to configure various coder attributes such as bit rate.

➢ **To configure codec attributes:**

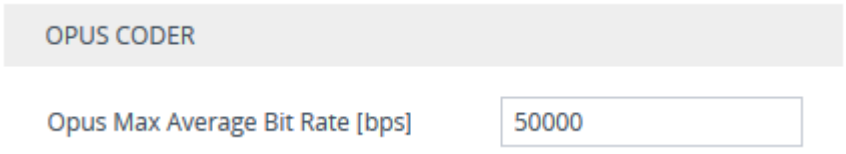
1. Open the Coder Settings page (**Setup** menu > **Signaling & Media** tab > **Coders & Profiles** folder > **Coder Settings**).
2. Configure the following parameters:
   - AMR coder:
     - 'AMR Payload Format' (AmrOctetAlignedEnable): Defines the AMR payload format type:



   - SILK coder (Skype's default audio codec):
     - 'SILK Tx Inband FEC': Enables forward error correction (FEC) for the SILK coder.
     - 'SILK Max Average Bit Rate': Defines the maximum average bit rate for the SILK coder.



   - Opus coder:
     - 'Opus Max Average Bitrate' (OpusMaxAverageBitRate): Defines the maximum average bit rate (in bps) for the Opus coder.



3. Click **Apply**.

Configuring Allowed Audio Coder Groups

The Allowed Audio Coders Groups table lets you configure up to 20 Allowed Audio Coders Groups. For each Allowed Audio Coders Group, you can configure up to 10 audio coders. The coders can include pre-defined coders and user-defined (string) coders for non-standard or unknown coders.

Allowed Audio Coders Groups restrict coders for SIP entities. Only coders listed in the Allowed Audio Coders Group (i.e., allowed coders) that is associated with the SIP entity can be used. If the coders in the SDP offer ('a=rtpmap' field) of the incoming SIP message are not listed in the Allowed Audio Coders Group, the device rejects the calls, unless transcoding is configured, whereby "extension" coders are added to the SDP, as described in Coder Transcoding. If the SDP offer contains some coders that are listed in the Allowed Audio Coders Group, the device manipulates the SDP offer by removing the coders that are not listed in the Allowed Audio Coders Group, before routing the SIP message to its destination. Thus, only coders that are common between the coders in the SDP offer and the coders in the Allowed Audio Coders Group are used. For more information on coder restriction, see Restricting Audio Coders.

For example, assume the following:

- The SDP offer in the incoming SIP message contains the G.729, G.711, and G.723 coders.
- The allowed coders configured for the SIP entity include G.711 and G.729.

The device removes the G.723 coder from the SDP offer, re-orders the coder list so that G.711 is listed first, and sends the SIP message containing only the G.711 and G.729 coders in the SDP.

To apply an Allowed Audio Coders Group for restricting coders to a SIP entity:

1. Configure an Allowed Audio Coders Group in the Allowed Audio Coders Groups table (see description below).
2. In the IP Profile associated with the SIP entity (see Configuring IP Profiles):
   - Assign the Allowed Audio Coders Group (using the IpProfile_SBCAllowedAudioCodersGroupName parameter).
   - Enable the use of Allowed Audio Coders Groups (by configuring the IpProfile_SBCAllowedCodersMode parameter to **Restriction** or **Restriction and Preference**).

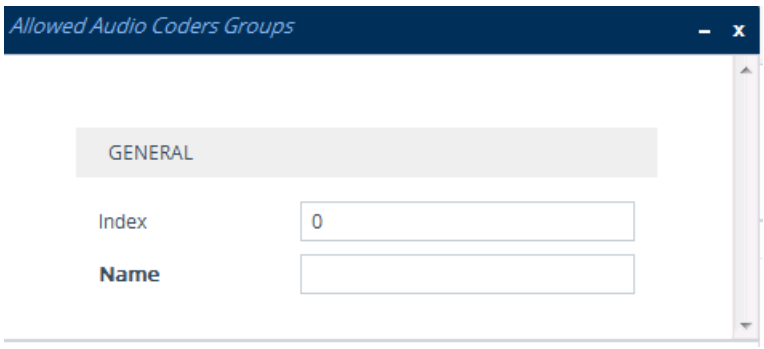
The device also re-orders (prioritizes) the coder list in the SDP according to the order of appearance of the coders listed in the Allowed Audio Coders Group. The first listed coder has the highest priority and the last coder has the lowest priority. For more information, see Prioritizing Coder List in SDP Offer.

- The Allowed Audio Coders Group for coder restriction takes precedence over the Coder Group for extension coders. In other words, if an extension coder is not listed as an allowed coder, the device does not add the extension coder to the SDP offer.
- To configure "extension" coders for adding to the SDP offer for audio transcoding, use the Coder Groups table (see Configuring Coder Groups

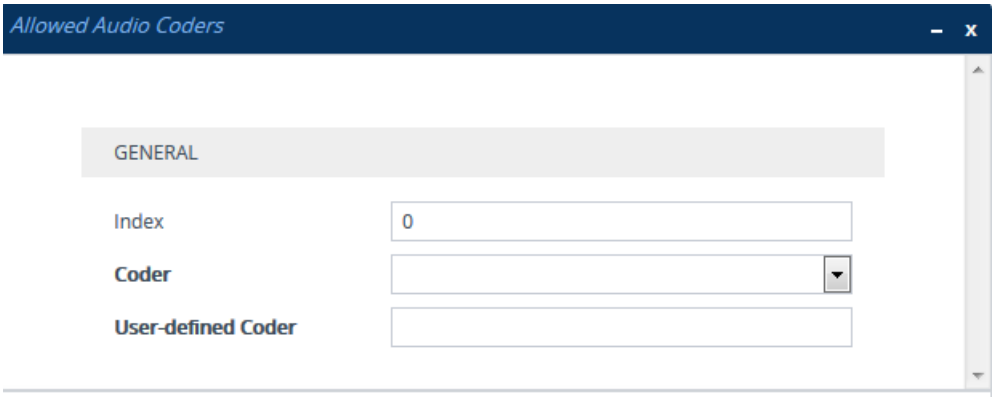
The following procedure describes how to configure Allowed Audio Coders Groups through the Web interface. You can also configure it through ini file [AllowedAudioCodersGroups] and [AllowedAudioCoders] or CLI (`configure voip > coders-and-profiles allowed-audio-coders-groups`; `configure voip > coders-and-profiles allowed-audio-coders < group index/coder index>`).

➢ **To configure an Allowed Audio Coders Group:**

1. Open the Allowed Audio Coders Groups table (**Setup** menu > **Signaling & Media** tab > **Coders & Profiles** folder > **Allowed Audio Coders Groups**).
2. Click **New**; the following dialog box appears:



3. Configure a name for the Allowed Audio Coders Group according to the parameters described in the table below.
4. Click **Apply**.
5. Select the new row that you configured, and then click the **Allowed Audio Coders** link located below the table; the Allowed Audio Coders table opens.
6. Click **New**; the following dialog box appears:



7. Configure coders for the Allowed Audio Coders Group according to the parameters described in the table below.
8. Click **Apply**.

Table 19-3: Allowed Audio Coders Groups and Allowed Audio Coders Tables Parameter Descriptions

Parameter	Description
Allowed Audio Coders Groups Table	
'Index' allowed-audio-coders-	Defines an index number for the new table row. **Note:** Each row must be configured with a unique

Parameter	Description
`groups <index>` [AllowedAudioCodersGroups_Index]	index.
'Name' `coders-group-name` [AllowedAudioCodersGroups_Name]	Defines a descriptive name, which is used when associating the row in other tables. The valid value is a string of up to 40 characters. **Note:** ■ Each row must be configured with a unique name. ■ The parameter value cannot contain a forward slash (/).
Allowed Audio Coders Table	
'Index' `allowed-audio-coders <group index/coder index>` [AllowedAudioCoders_AllowedAudioCodersIndex]	Defines an index number for the new table row. For a list of supported coders, see **Note:** Each row must be configured with a unique index.
'Coder' `coder` [AllowedAudioCoders_CoderID]	Defines a coder from the list of coders. **Note:** Each coder can be configured only once per Allowed Audio Coders Group.
'User-defined Coder' `user-define-coder` [AllowedAudioCoders_UserDefineCoder]	Defines a user-defined coder. The valid value is a string of up to 24 characters (case-insensitive). For example, "HD.123" (without quotation marks). **Note:** Each coder can be configured only once per Allowed Audio Coders Group.

Configuring Allowed Video Coder Groups

The Allowed Video Coders Groups table lets you configure up to four Allowed Video Coders Groups for SBC calls. Each Allowed Video Coders Group can be configured with up to 10 user-defined (string) video coders. An Allowed Video Coders Group defines a list of video coders that can be used when forwarding video streams to a specific SIP entity.

Allowed Video Coders Groups are assigned to SIP entities, using IP Profiles (see Configuring IP Profiles). The video coders appear in the SDP media type "video" ('m=video' line). Coders that are not listed in the Allowed Video Coders Group are removed from the SDP offer that is sent to

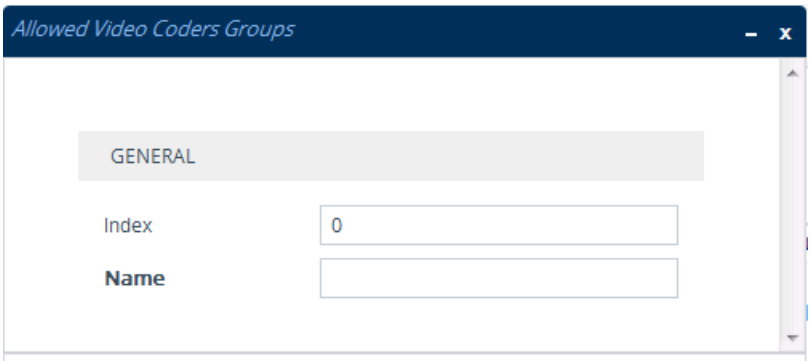
the SIP entity. Only coders that are common between the coders in the SDP offer and the coders listed in the Allowed Video Coders Group are used. Thus, Allowed Video Coders Groups enable you to enforce the use of only specified coders. For more information, see Restricting Audio Coders.

The order of appearance of the coders listed in the Allowed Video Coders Group determines the priority (preference) of the coders in the SDP offer. The device arranges the SDP offer's coder list according to their order in the Allowed Video Coders Group. The priority is in descending order, whereby the first coder in the list is given the highest priority and the last coder, the lowest priority. For more information, see Prioritizing Coder List in SDP Offer.

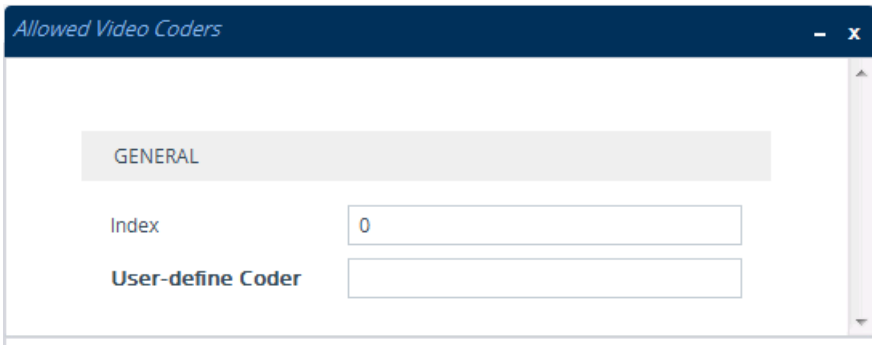
The following procedure describes how to configure Allowed Video Coders Groups through the Web interface. You can also configure it through ini file [AllowedVideoCodersGroups] and [AllowedVideoCoders] or CLI (`configure voip > coders-and-profiles allowed-video-coders-groups`; `configure voip > coders-and-profiles allowed-video-coders < group index/coder index`>).

➢ To configure an Allowed Video Coders Group:

1. Open the Allowed Video Coders Groups table (**Setup** menu > **Signaling & Media** tab > **Coders & Profiles** folder > **Allowed Video Coders Groups**).
2. Click **New**; the following dialog box appears:



3. Configure a name for the Allowed Video Coders Group according to the parameters described in the table below.
4. Click **Apply**.
5. Select the new row that you configured, and then click the **Allowed Video Coders** link located below the table; the Allowed Video Coders table opens.
6. Click **New**; the following dialog box appears:



7. Configure coders for the Allowed Video Coders Group according to the parameters described in the table below.
8. Click **Apply**.

Table 19-4: Allowed Video Coders Groups and Allowed Video Coders Tables Parameter Descriptions

Parameter	Description
Allowed Video Coders Groups Table	
'Index' `allowed-video-coders-groups <index>` [AllowedVideoCodersGroups_Index]	Defines an index number for the new table row. **Note:** Each row must be configured with a unique index.
'Name' `coders-group-name` [AllowedVideoCodersGroups_Name]	Defines a descriptive name, which is used when associating the row in other tables. The valid value is a string of up to 40 characters. **Note:** ■ Each row must be configured with a unique name. ■ The parameter value cannot contain a forward slash (/).
Allowed Video Coders Table	
'Index' `allowed-video-coders <group index/coder index>` [AllowedVideoCoders_AllowedVideoCodersIndex]	Defines an index number for the new table row. **Note:** Each row must be configured with a unique index.
'User Define Coder' `user-define-coder` [AllowedVideoCoders_	Defines a user-defined coder. The valid value is a string of up to 24 characters (case-insensitive). For example, "HD.123" (without quotation

Parameter	Description
UserDefineCoder]	marks). **Note:** Each coder can be configured only once per Allowed Video Coders Group.

Configuring IP Profiles

The IP Profiles table lets you configure up to 125 IP Profiles. An IP Profile is a set of parameters with user-defined settings relating to signaling (e.g., SIP message terminations such as REFER) and media (e.g., coder type). An IP Profile can later be assigned to specific IP calls (inbound and/or outbound). Thus, IP Profiles provide high-level adaptation when the device interworks between different SIP user agents (UA), each of which may require different handling by the device. This can include, for example, transcoding or even transrating (of packetization time). For example, if a specific SIP UA uses the G.711 coder only, you can configure an IP Profile with G.711 for this UA.

Many of the parameters in the IP Profiles table have a corresponding "global" parameter, whose settings apply to all calls that are not associated with an IP Profile. The default value of these IP Profile parameters is the same as the default value of their corresponding global parameters. However, if you change a global parameter from its default value, the value of its corresponding IP Profile parameter inherits its value for all subsequently created (new) IP Profiles. For example, the IP Profile parameter for configuring maximum call duration is [IpProfile_SBCMaxCallDuration]. Its corresponding global parameter is [SBCMaxCallDuration]. The default of the global parameter is "0" and therefore, the default of this IP Profile parameter is also "0". However, if you configure the global parameter to "10", the value of this IP Profile parameter for all subsequently created (new) IP Profiles is also "10".

To use your IP Profile for specific calls, you need to assign it to any of the following:

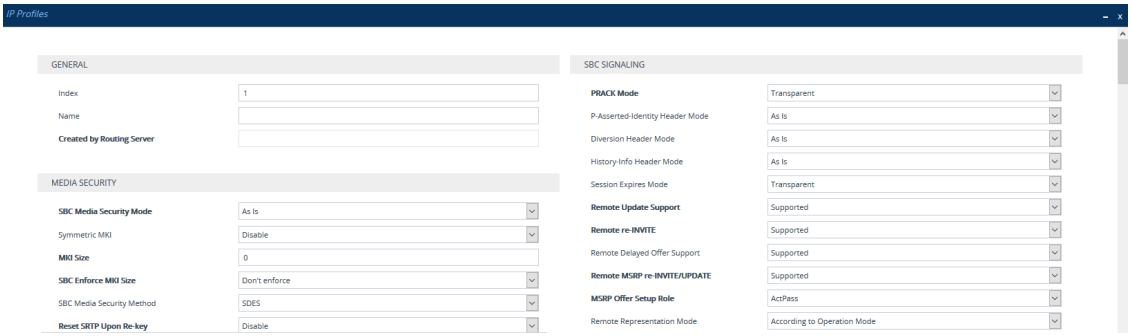
- IP Groups - see Configuring IP Groups

> You can also use IP Profiles when using a Proxy server (when the AlwaysUseRouteTable parameter is set to 1).

The following procedure describes how to configure IP Profiles through the Web interface. You can also configure it through ini file [IPProfile] or CLI (`configure voip > coders-and-profiles ip-profile`).

➢ To configure an IP Profile:

1. Open the IP Profiles table (**Setup** menu > **Signaling & Media** tab > **Coders & Profiles** folder > **IP Profiles**).
2. Click **New**; the following dialog box appears:



3. Configure an IP Profile according to the parameters described in the table below.
4. Click **Apply**.

Table 19-5: IP Profiles Table Parameter Descriptions

Parameter	Description
General	
'Index' [IpProfile_Index]	Defines an index number for the new table row. **Note:** Each row must be configured with a unique index.
'Name' `profile-name` [IpProfile_ProfileName]	Defines a descriptive name, which is used when associating the row in other tables. The valid value is a string of up to 40 characters. **Note:** The parameter value cannot contain a forward slash (/).
'Created By Routing Server' [IpProfile_ CreatedByRoutingServer]	(Read-only) Indicates whether the IP Profile was created by a third-party routing server: ■ [0] **No** ■ [1] **Yes** For more information on the third-party routing server feature, see Centralized Third-Party Routing Server.
Media Security	
'SBC Media Security Mode' `sbc-media-security-behaviour` [IpProfile_ SBCMediaSecurityBehaviour]	Defines the handling of RTP/SRTP, and MSRP/MSRPS for the SIP UA associated with the IP Profile. ■ [0] **As is** = (Default) No special handling for RTP/SRTP and MSRP/MSRPS is done. ■ [1] **Secured** = SBC legs negotiate only SRTP/MSRPS media lines, and RTP/MSRP media lines are removed from the incoming SDP offer-answer. ■ [2] **Not Secured** = SBC legs negotiate only RTP/MSRP

Parameter	Description
	media lines, and SRTP/MSRPS media lines are removed from the incoming offer-answer. ■ [3] **Both** = Each offer-answer is extended (if not already) to two media lines - one RTP/MSRPand the other SRTP/MSRPS. If two SBC legs (after offer-answer negotiation) use different security types (i.e., one RTP/MSRP and the other SRTP/MSRPS), the device performs RTP-SRTP/MSRP-MSRPS transcoding. For such transcoding, the following prerequisites must be met: ■ At least one supported SDP "crypto" attribute and parameters. ■ The [EnableMediaSecurity] parameter must be configured to [1]. If one of the above prerequisites is not met, then: ■ any value other than **As is** is discarded. ■ if the incoming offer is SRTP/MSRPS, forced transcoding, coder transcoding, and DTMF extensions are not applied. **Note:** For secured MSRP (MSRPS), configure the parameter to **Secured** or **Both**. For more information on MSRP, see Configuring Message Session Relay Protocol on page 779.
'Symmetric MKI' `enable-symmetric-mki` [IpProfile_ EnableSymmetricMKI]	Enables symmetric MKI negotiation. ■ [0] **Disable** = (Default) The device includes the MKI in its SIP 200 OK response according to the SRTPTxPacketMKISize parameter (if set to 0, it is not included; if set to any other value, it is included with this value). ■ [1] **Enable** = The answer crypto line contains (or excludes) an MKI value according to the selected crypto line in the offer. For example, assume that the device receives an INVITE containing the following two crypto lines in SDP: `a=crypto:2 AES_CM_128_HMAC_SHA1_80 inline:TAaxNnQt8/qLQMnDuG4vxYfWl6K7eBK/ ufk04pR4|2^31|1:1`

Parameter	Description

	`a=crypto:3 AES_CM_128_HMAC_SHA1_80 inline:bnuYZnMxSfUiGitviWJZmzr7OF3AiRO0l5Vnh0kH|2^31`
	The first crypto line includes the MKI parameter "1:1". In the 200 OK response, the device selects one of the crypto lines (i.e., '2' or '3'). Typically, it selects the first line that supports the crypto suite. However, for SRTP-to-SRTP in SBC sessions, it can be determined by the remote side on the outgoing leg. If the device selects crypto line '2', it includes the MKI parameter in its answer SDP, for example:
	`a=crypto:2 AES_CM_128_HMAC_SHA1_80 inline:R1VyA1xV/qwBjkEklu4kSJyl3wCtYeZLq1/QFuxw|2^31|1:1`
	If the device selects a crypto line that does not contain the MKI parameter, then the MKI parameter is not included in the crypto line in the SDP answer (even if the SRTPTxPacketMKISize parameter is set to any value other than 0).
	Note: The corresponding global parameter is EnableSymmetricMKI.

'MKI Size' `mki-size` [IpProfile_MKISize]	Defines the size (in bytes) of the Master Key Identifier (MKI) in SRTP Tx packets. The valid value is 0 to 4. The default is 0 (i.e., new keys are generated without MKI). **Note:** ■ The device can forward MKI size as is for SRTP-to-SRTP flows or override the MKI size during negotiation. This can be done on the inbound or outbound leg. ■ The corresponding global parameter is SRTPTxPacketMKISize.
'SBC Enforce MKI Size' `sbc-enforce-mki-size` [IpProfile_ SBCEnforceMKISize]	Enables negotiation of the Master Key Identifier (MKI) length for SRTP-to-SRTP flows between SIP networks (i.e., IP Groups). This includes the capability of modifying the MKI length on the inbound or outbound SBC call leg for the SIP UA associated with the IP Profile. ■ [0] **Don't enforce** = (Default) Device forwards the MKI size as is.

Parameter	Description
	■ [1] **Enforce** = Device changes the MKI length according to the settings of the IP Profile parameter, MKISize.
'SBC Media Security Method' `sbc-media-security-method` [IpProfile_SBCMediaSecurityMethod]	Defines the media security protocol for SRTP, for the SIP UA associated with the IP Profile. ■ [0] **SDES** = (Default) The device secures RTP using the Session Description Protocol Security Descriptions (SDES) protocol to negotiate the cryptographic keys (RFC 4568). The keys are sent in the SDP body ('a=crypto') of the SIP message and are typically secured using SIP over TLS (SIPS). The encryption of the keys is in plain text in the SDP. SDES implements TLS over TCP. ■ [1] **DTLS** = The device uses Datagram Transport Layer Security (DTLS) protocol to secure UDP-based media streams (RFCs 5763 and 5764). For more information on DTLS, see SRTP using DTLS Protocol. ■ [2] **Both** = SDES and DTLS protocols are supported. **Note:** ■ To support DTLS, you must also configure the following for the SIP UA: ✓ TLS Context for DTLS (see Configuring TLS Certificate Contexts). The server cipher ('Cipher Server') must be configured to All. ✓ IpProfile_SBCMediaSecurityBehaviourMedia configured to SRTP or Both. ✓ IpProfile_SBCRTCPMux configured to Supported. The setting is required as the DTLS handshake is done for the port used for RTP. Therefore, RTCP and RTP should be multiplexed over the same port. ■ The device does not support forwarding of DTLS transparently between SIP UAs. ■ As DTLS has been defined by the WebRTC standard as mandatory for encrypting media channels for SRTP key exchange, the support is important for deployments implementing WebRTC. For more information on WebRTC, see WebRTC.

Parameter	Description
'Reset SRTP Upon Re-key' reset-srtp-upon-re-key [IpProfile_ ResetSRTPStateUponRekey]	Enables synchronization of the SRTP state between the device and a server when a new SRTP key is generated upon a SIP session expire. This feature ensures that the roll-over counter (ROC), one of the parameters used in the SRTP encryption/decryption process of the SRTP packets is synchronized on both sides for transmit and receive packets. ■ [0] **Disable** = (Default) ROC is not reset on the device side. ■ [1] **Enable** = If the session expires causing a session refresh through a re-INVITE, the device or server generates a new key and the device resets the ROC index (and other SRTP fields) as done by the server, resulting in a synchronized SRTP. **Note:** ■ If this feature is disabled and the server resets the ROC upon a re-key generation, one-way voice may occur. ■ The corresponding global parameter is [ResetSRTPStateUponRekey].
'Generate SRTP Keys Mode' generate-srtp-keys [IpProfile_ GenerateSRTPKeys]	Enables the device to generate a new SRTP key upon receipt of a re-INVITE with the SIP UA associated with the IP Profile. ■ [0] **Only If Required** = (Default) The device generates an SRTP key only if necessary. ■ [1] **Always** = The device always generates a new SRTP key.
'SBC Remove Crypto Lifetime in SDP' sbc-sdp-remove-crypto-lifetime [IpProfile_ SBCRemoveCryptoLifetimeIn SDP]	Defines the handling of the lifetime field in the 'a=crypto' attribute of the SDP for the SIP UA associated with the IP Profile. The SDP field defines the lifetime of the master key as measured in maximum number of SRTP or SRTCP packets using the master key. ■ [0] **No** = (Default) The device retains the lifetime field (if present) in the SDP. ■ [1] **Yes** = The device removes the lifetime field from the 'a=crypto' attribute. **Note:** If you configure the parameter to Yes, the following IP Profile parameters must be configured as follows:

Parameter	Description
	■ IpProfile_EnableSymmetricMKI configured to Enable [1]. ■ IpProfile_MKISize configured to 0. ■ IpProfile_SBCEnforceMKISize configured to Enforce [1].
'SBC Remove Unknown Crypto' sbc-remove-unknown-crypto [IpProfile_SBCRemoveUnKnownCrypto]	Defines whether the device keeps or removes unknown cryptographic suites (encryption and authentication algorithms) that are present in the SDP 'a=crypto' attribute in the incoming SIP message, before forwarding the message to the SIP UA associated with this IP Profile. ■ [0] **No** = (Default) The device keeps all unknown cryptographic suites that are in the SDP's 'a=crypto' attribute. ■ [1] **Yes** = The device removes all unknown cryptographic suites that are in the SDP's 'a=crypto' attribute. **Note:** ■ The feature is applicable only to SRTP-to-SRTP calls and calls that do not require transcoding.
SBC Early Media	
'Remote Early Media' sbc-rmt-early-media-supp [IpProfile_SBCRemoteEarlyMediaSupport]	Defines whether the remote side can accept early media or not. ■ [0] **Not Supported** = Early media is not supported. ■ [1] **Supported** = (Default) Early media is supported.
'Remote Multiple 18x' sbc-rmt-mltple-18x-supp [IpProfile_SBCRemoteMultiple18xSupport]	Defines whether multiple 18x responses including 180 Ringing, 181 Call is Being Forwarded, 182 Call Queued, and 183 Session Progress are forwarded to the caller, for the SIP UA associated with the IP Profile. ■ [0] **Not Supported** = Only the first 18x response is forwarded to the caller. ■ [1] **Supported** = (Default) Multiple 18x responses are forwarded to the caller.
'Remote Early Media	Defines the SIP provisional response type - 180 or 183 -

Parameter	Description
Response Type' sbc-rmt-early-media-resp [IpProfile_SBCRemoteEarlyMediaResponseType]	for forwarding early media to the caller, for the SIP UA associated with the IP Profile. ■ [0] **Transparent** = (Default) All early media response types are supported; the device forwards all responses as is (unchanged). ■ [1] **180** = Early media is sent as 180 response only. ■ [2] **183** = Early media is sent as 183 response only.

'Remote Multiple Early Dialogs'
sbc-multi-early-diag
[IpProfile_SBCRemoteMultipleEarlyDialogs]
	Defines the device's handling of To-header tags in call forking responses (i.e., multiple SDP answers) sent to the SIP UA associated with the IP Profile.
	When the SIP UA initiates an INVITE that is subsequently forked (for example, by a proxy server) to multiple UAs, the endpoints respond with a SIP 183 containing an SDP answer. Typically, each endpoint's response has a different To-header tag. For example, a call initiated by the SIP UA (100@A) is forked and two endpoints respond with ringing, each with a different tag:
	■ Endpoint "tag 2":
	`SIP/2.0 180 Ringing`
	`From: <sip:100@A>;tag=tag1`
	`To: sip:200@B;tag=tag2`
	`Call-ID: c2`
	■ Endpoint "tag 3":
	`SIP/2.0 180 Ringing`
	`From: <sip:100@A>;tag=tag1`
	`To: sip:200@B;tag=tag3`
	`Call-ID: c2`
	In non-standard behavior (when the parameter is configured to Disable), the device forwards all the SDP answers with the same tag. In the example, endpoint "tag 3" is sent with the same tag as endpoint "tag 2" (i.e., To: sip:200@B;tag=tag2).
	■ [-1] **According to Operation Mode** = (Default) Depends on the setting of the 'Operation Mode' parameter in the IP Group or SRDs table:
	✓ B2BUA: Device operates as if the parameter is set to Disable [0].

Parameter	Description
	✓ Call State-full Proxy: Device operates as if the parameter is set to Enable [1]. In addition, the device preserves the From tags and Call-IDs of the endpoints in the SDP answer sent to the SIP UA. ■ [0] **Disable** = Device sends the multiple SDP answers with the same To-header tag, to the SIP UA. In other words, this option is relevant if the SIP UA does not support multiple dialogs (and multiple tags). However, non-standard, multiple answer support may still be configured by the SBCRemoteMultipleAnswersMode parameter. ■ [1] **Enable** = Device sends the multiple SDP answers with different To-header tags, to the SIP UA. In other words, the SIP UA supports standard multiple SDP answers (with different To-header tags). In this case, the SBCRemoteMultipleAnswersMode parameter is ignored. **Note:** If the parameter and the SBCRemoteMultipleAnswersMode parameter are disabled, multiple SDP answers are not reflected to the SIP UA (i.e., the device sends the same SDP answer in multiple 18x and 200 responses).
'Remote Multiple Answers Mode' `sbc-multi-answers` [IpProfile_ SBCRemoteMultipleAnswersMode]	Enables interworking multiple SDP answers within the same SIP dialog (non-standard). The parameter enables the device to forward multiple answers to the SIP UA associated with the IP Profile. The parameter is applicable only when the IpProfile_SBCRemoteMultipleEarlyDialogs parameter is disabled. ■ [0] **Disable** = (Default) Device always sends the same SDP answer, which is based on the first received answer that it sent to the SIP UA, for all forked responses (even if the 'Forking Handling Mode' parameter is **Sequential**), and thus, may result in transcoding. ■ [1] **Enable** = If the 'Forking Handling Mode' parameter is configured to **Sequential**, the device sends multiple SDP answers.
'Remote Early Media RTP Detection Mode'	Defines whether the destination UA sends RTP immediately after it sends a 18x response.

Parameter	Description
sbc-rmt-early-media-rtp [IpProfile_SBCRemoteEarlyMediaRTP]	■ [0] **By Signaling** = (Default) Remote client sends RTP immediately after it sends 18x response with early media. The device forwards 18x and RTP as is. ■ [1] **By Media** = After sending 18x response, the remote client waits before sending RTP (e.g., Microsoft Skype for Business environment). For the device's handling of this remote UA support, see Interworking SIP Early Media.
'Remote RFC 3960 Support' sbc-rmt-rfc3960-supp [IpProfile_SBCRemoteSupportsRFC3960]	Defines whether the destination UA is capable of receiving 18x messages with delayed RTP. ■ [0] **Not Supported** = (Default) UA does not support receipt of 18x messages with delayed RTP. For the device's handling of this remote UA support, see Interworking SIP Early Media. ■ [1] **Supported** = UA is capable of receiving 18x messages with delayed RTP.
'Remote Can Play Ringback' sbc-rmt-can-play-ringback [IpProfile_SBCRemoteCanPlayRingback]	Defines whether the destination UA can play a local ringback tone. ■ [0] **No** = UA does not support local ringback tone. The device sends 18x with delayed SDP to the UA. ■ [1] **Yes** = (Default) UA supports local ringback tone. For the device's handling of this remote UA support, see Interworking SIP Early Media.
'Generate RTP' sbc-generate-rtp [IPProfile_SBCGenerateRTP]	Enables the device to generate "silence" RTP packets to the SIP UA until it detects audio RTP packets from the SIP UA. The parameter provides support for interworking with SIP entities that wait for the first incoming packets before sending RTP (e.g., early media used for ringback tone or IVR) during media negotiation. ■ [0] **None** = (Default) Silence packets are not generated. ■ [1] **Until RTP Detected** = The device generates silence RTP packets to the SIP UA upon receipt of a SIP response (183 with SDP) from the SIP UA. In other words, these packets serve as the first incoming packets for the SIP UA. The device stops sending silence packets when it receives RTP packets from the

Parameter	Description
	peer side (which it then forwards to the SIP UA). **Note:** To generate silence packets, DSP resources are required (except for calls using the G.711 coder).
SBC Media	
'Mediation Mode' `transcoding-mode` [IpProfile_TranscodingMode]	Defines the transcoding mode (media negotiation) for the SIP UA associated with the IP Profile. ■ [0] **RTP Mediation** = (Default) Transcoding is done only if required. If not required, many of the media settings (such as gain control) are not applied to the voice stream. The device forwards the RTP packets transparently (i.e., RTP-to-RTP) without processing the data; only the RTP headers are re-constructed. ■ [1] **Force Transcoding** = This enables the device to receive capabilities that are not negotiated between the SIP entities, by implementing DSP transcoding. For example, it can enforce gain control to use voice transcoding even though both legs have negotiated without the device's intervention (such as Extension coders). ■ [2] **RTP Forwarding** = If transcoding is not required and both legs are configured with RTP forwarding, then RTP packets are forwarded transparently without any processing. This mode is needed when the call parties pass invalid RTP packets on the RTP port. If you use this option, you may also need to configure the global parameters 'Forward Unknown RTP Payload Types' to **Handle as Valid Packet**, and 'Forward Invalid RTP Packets' to **Forward Packets**. **Note:** ■ For transcoding, make sure that the device's License Key includes a license for the number of DSP resources ('DSP Channels') and a license for the number of transcoding sessions ('Transcoding Sessions'). For more information on the License Key, see License Key on page 871. ■ Each transcoding session uses two DSP resources. ■ The corresponding global parameter is [TranscodingMode].

Parameter	Description
'Extension Coders Group' sbc-ext-coders-group-name [IpProfile_SBCExtensionCodersGroupName]	Assigns a Coder Group for extension coders, which are added to the SDP offer in the outgoing leg for the SIP UA associated with the IP Profile. This is used when transcoding is required between two IP entities (i.e., the SDP answer from one doesn't include any coder included in the offer previously sent by the other). For more information on extension coders and transcoding, see Coder Transcoding. To configure Coder Groups, see Configuring Coder Groups.
'Allowed Audio Coders' allowed-audio-coders-group-name [IpProfile_SBCAllowedAudioCodersGroupName]	Assigns an Allowed Audio Coders Group, which defines audio (voice) coders that can be used for the SIP UA associated with the IP Profile. To configure Allowed Audio Coders Groups, see Configuring Allowed Audio Coder Groups. For a description of the Allowed Coders feature, see Restricting Coders.
'Allowed Coders Mode' sbc-allowed-coders-mode [IpProfile_SBCAllowedCodersMode]	Defines the mode of the Allowed Coders feature for the SIP UA associated with the IP Profile. ■ [0] **Restriction** = In the incoming SDP offer, the device uses only Allowed coders; the rest are removed from the SDP offer (i.e., only coders common between those in the received SDP offer and the Allowed coders are used). If an Extension Coders Group is also assigned (using the 'Extension Coders Group' parameter, above), these coders are added to the SDP offer if they also appear in Allowed coders. ■ [1] **Preference** = The device re-arranges the priority (order) of the coders in the incoming SDP offer according to their order of appearance in the Allowed Audio Coders Group or Allowed Video Coders Group. The coders in the original SDP offer are listed after the Allowed coders. ■ [2] **Restriction and Preference** = Performs both Restriction and Preference. **Note:** ■ The parameter is applicable only if Allowed coders are assigned to the IP Profile (see the 'Allowed Audio Coders' or 'Allowed Video Coders' parameters).

Parameter	Description
	■ For more information on the Allowed Coders feature, see Restricting Coders.
'Allowed Video Coders' allowed-video-coders-group-name [IpProfile_SBCAllowedVideoCodersGroupName]	Assigns an Allowed Video Coders Group. This defines permitted video coders when forwarding video streams to the SIP UA associated with the IP Profile. The video coders are listed in the "video" media type in the SDP (i.e., 'm=video' line). For this SIP UA, the device uses only video coders that appear in both the SDP offer and the Allowed Video Coders Group. By default, no Allowed Video Coders Group is assigned (i.e., all video coders are allowed). To configure Allowed Video Coders Groups, see Configuring Allowed Video Coder Groups.
'Allowed Media Types' sbc-allowed-media-types [IpProfile_SBCAllowedMediaTypes]	Defines media types permitted for the SIP UA associated with the IP Profile. The media type appears in the SDP 'm=' line (e.g., 'm=audio'). The device permits only media types that appear in both the SDP offer and this configured list. If no common media types exist between the SDP offer and this list, the device drops the call. The valid value is a string of up to 64 characters. To configure multiple media types, separate the strings with a comma, e.g., " audio, text" (without quotation marks). By default, no media types are configured (i.e., all media types are permitted).
'Direct Media Tag' sbc-dm-tag [IPProfile_SBCDirectMediaTag]	Defines an identification tag for enabling direct media or media bypass (i.e., no Media Anchoring) of SBC calls for the SIP UA associated with the IP Profile. Direct media occurs between all UAs whose IP Profiles have the same tag value (non-empty value). For example, if you configure the parameter to "direct-rtp" for two IP Profiles "IP-PBX-1" and "IP-PBX-2", the device employs direct media for calls of UAs associated with IP Profile "IP-PBX-1", for calls of UAs associated with IP Profile "IP-PBX-2", and for calls between UAs associated with IP Profile "IP-PBX-1" and IP Profile "IP-PBX-2". The valid value is a string of up to 16 characters. By default, no value is defined. For more information on direct media, see Direct Media. **Note:**

Parameter	Description
	■ If you enable direct media for the IP Profile, make sure that your Media Realm provides enough ports, as media may traverse the device for mid-call services (e.g., call transfer). ■ If you have configured a SIP Recording rule (see SIP-based Media Recording on page 247) for calls associated with this IP Profile, the device automatically disables direct media for these calls (during their SIP signaling setup). This ensures that the media passes through the device so that it can be recorded and sent to the SRS. However, if you enable direct media using the [SBCDirectMedia] global parameter (i.e., for all calls), direct media is always enforced and calls will not be recorded.
'RFC 2833 Mode' `sbc-rfc2833-behavior` [IpProfile_ SBCRFC2833Behavior]	Defines the handling of RFC 2833 SDP offer-answer negotiation for the SIP UA associated with the IP Profile. ■ [0] **As is** = (Default) The device does not intervene in the RFC 2833 negotiation. ■ [1] **Extend** = Each outgoing offer-answer includes RFC 2833 in the offered SDP. The device adds RFC 2833 only if the incoming offer does not include RFC 2833. ■ [2] **Disallow** = The device removes RFC 2833 from the incoming offer. **Note:** If the device interworks between different DTMF methods and one of the methods is in-band DTMF packets (in RTP), detection and generation of DTMF methods requires DSP resources. However, RFC 2833 to SIP INFO does not require DSP resources.
'RFC 2833 DTMF Payload Type' `sbc-2833dtmf-payload` [IpProfile_ SBC2833DTMFPayloadType]	Defines the payload type of DTMF digits for the SIP UA associated with the IP Profile. This enables the interworking of the DTMF payload type for RFC 2833 between different SBC call legs. For example, if two entities require different DTMF payload types, the SDP offer received by the device from one UA is forwarded to the destination UA with its payload type replaced with the configured payload type, and vice versa. The value range is 0 to 200. The default is 0 (i.e., the device forwards the received payload type as is).

Parameter	Description
'Alternative DTMF Method' sbc-alternative-dtmf-method [IpProfile_SBCAlternativeDTMFMethod]	The device's first priority for DTMF method at each leg is RFC 2833. Thus, if the device successfully negotiates RFC 2833 for the SIP UA associated with the IP Profile, the chosen DTMF method for this leg is RFC 2833. When RFC 2833 negotiation fails, the device uses the DTMF method configured by this parameter for the leg. ■ [0] **As Is** = (Default) The device does not attempt to interwork any special DTMF method. ■ [1] **In Band** ■ [2] **INFO - Cisco** ■ [3] **INFO - Nortel** ■ [4] **INFO - Lucent = INFO, Korea** **Note:** If the device interworks between different DTMF methods and one of the methods is in-band DTMF packets (in RTP), detection and generation of DTMF methods requires DSP resources. However, RFC 2833 to SIP INFO does not require DSP resources.
'Send Multiple DTMF Methods' sbc-send-multiple-dtmf-methods [IPProfile_SBCSupportMultipleDTMFMethods]	Enables the device to send DTMF digits out-of-band (not with audio stream) using both the SIP INFO and RFC 2833 methods for the same call on the leg to which this IP Profile is associated. The RFC 2833 method sends out-of-band DTMF digits using the RTP protocol while the SIP INFO method sends the digits using the SIP protocol. ■ [0] **Disable** = (Default) The device sends DTMF digits using only one method (either SIP INFO, RFC 2833, or in-band). ■ [1] **Enable** = The device sends DTMF digits using both methods - SIP INFO and RFC 2833. If you have enabled the parameter, you can also configure the device to stop sending DTMF digits using the SIP INFO method if the device receives a SIP re-INVITE (or UPDATE) from the SIP UA to where the SIP INFO is being sent (and keep sending the DTMF digits using the RFC 2833 method). This is done using the AudioCodes proprietary SIP header X-AC-Action and a Message Manipulation rule (inbound) to instruct the device to switch to a different IP Profile that is configured to disable the sending of DTMF digits using both methods (i.e., 'Send Multiple DTMF

Parameter	Description

	Methods' is configured to Disable):
	`X-AC-Action: 'switch-profile;profile-name=<IP Profile Name>'`
	If the IP Profile name contains one or more spaces, you must enclose the name in double quotation marks, for example:
	`X-AC-Action: 'switch-profile;profile-name="My IP Profile"'`
	The Message Manipulation rule adds the proprietary header with the value of the new IP Profile to the incoming re-INVITE or UPDATE message and as a result, the device uses the new IP Profile for the SIP UA and stops sending it SIP INFO messages. You can also configure an additional Message Manipulation rule to re-start the sending of the SIP INFO. For example, you can configure two Message Manipulation rules where the sending of both SIP INFO and RFC 2833 depends on the negotiated media port -- the device stops sending SIP INFO if the SDP of the re-INVITE or UPDATE message contains port 7550 and re-starts sending if the port is 8660. The rule that re-starts the SIP INFO switches the IP Profile back to the initial IP Profile that enables the sending of DTMF digits using both methods (i.e., 'Send Multiple DTMF Methods' is configured to Enable). The configured Message Manipulation rules for this example are shown below:
	■ Index 1
	✓ Message Type: reinvite.request
	✓ Condition: body.sdp regex (.*)(m=audio 7550 RTP/AVP)(.*)
	✓ Action Subject: header.X-AC-Action
	✓ Action Type: Add
	✓ Action Value: 'switch-profile;profile-name=ITSP-Profile-2'
	■ Index 2
	✓ Message Type: reinvite.request
	✓ Condition: body.sdp regex (.*)(m=audio 8660 RTP/AVP)(.*)
	✓ Action Subject: header.X-AC-Action

Parameter	Description
	✓ Action Type: Add ✓ Action Value: 'switch-profile;profile-name=ITSP-Profile-1' The Message Manipulation rules must be assigned to the SIP UA's IP Group, using the 'Inbound Message Manipulation Set' parameter. **Note:** ■ To send DTMF digits using both methods (i.e., when the parameter is enabled), you need to also configure the following: ✓ Configure the 'Alternative DTMF Method' (IPProfile_SBCAlternativeDTMFMethod) parameter to one of the SIP INFO options (INFO – Cisco, INFO – Nortel, or INFO – Lucent). ✓ Enable the sending of DTMF digits using the RFC 2833 method, by configuring the 'RFC 2833 Mode' (IpProfile_SBCRFC2833Behavior) parameter to As Is or Extend. ■ When using the X-AC-Action header to switch IP Profiles, it is recommended that the settings of the switched IP Profile are identical (except for the 'Send Multiple DTMF Methods' parameter) to the initial IP Profile in order to avoid any possible call handling errors. ■ The parameter is applicable only to the SBC application.
'Receive Multiple DTMF Methods' `sbc-receive-multiple-dtmf-methods` [IpProfile_ReceiveMultipleDTMFMethods]	Enables the device to receive DTMF digits out-of-band (not with audio stream) using both the SIP INFO and RFC 2833 methods, but forwards the DTMF only using RFC 2833. ■ [0] **Disable** = (Default) The device receives DTMF digits only by the RFC 2833 method, if negotiated. Otherwise, the device uses the DTMF method according to the IP Profile's 'Alternative DTMF Method' parameter (see above). In other words, it receives DTMF digits ussing only one method only. ■ [1] **Enable** = The device receives DTMF digits using the SIP INFO message method even if both sides

Parameter	Description
	successfully negotiated the RFC 2833 method. In other words, both SIP INFO and RFC 2833 are used to detect DTMF digits by the device. However, the device forwards the DTMF using RFC 2833 only.
'Adapt RFC2833 BW to Voice coder BW' `sbc-adapt-rfc2833-bw-voice-bw` [IpProfile_SBCAdaptRFC2833BWToVoiceCoderBW]	Defines the 'telephone-event' type (8000 or 16000) in the SDP that the device sends in the outgoing SIP 200 OK message for DTMF payload negotiation (sampling rate). ■ [0] **Disable** = (Default) The device always sends the 'telephone-event' as 8000 in the outgoing SIP 200 OK, even if the SDP of the incoming INVITE contains multiple telephone-event types (e.g., 8000 and 16000). ■ [1] **Enable** = The type of 'telephone-event' that the device sends in the outgoing SIP 200 OK message is according to the coder type (narrowband or wideband). If narrowband, it sends the 'telephone-event' as 8000; if wideband, it sends it as 16000. An example when the parameter is configured to **Enable** is shown below, whereby the 'telephone-event' is "16000" in the outgoing message due to the wideband coder: **SDP in incoming INVITE:** `a=rtpmap:97 AMR-WB/16000/1 a=fmtp:97 mode-change-capability=2 a=rtpmap:98 AMR-WB/16000/1 a=fmtp:98 octet-align=1; mode-change-capability=2 a=rtpmap:100 AMR/8000/1 a=fmtp:100 mode-change-capability=2 a=rtpmap:99 telephone-event/16000/1 a=fmtp:99 0-15 a=rtpmap:102 telephone-event/8000/1 a=fmtp:102 0-15` **SDP in outgoing 200 OK:** `m=audio 6370 RTP/AVP 97 99 a=rtpmap:99 telephone-event/16000/1 a=fmtp:99 0-15 a=sendrecv a=ptime:20 a=maxptime:120 a=rtpmap:97 AMR-WB/16000 a=fmtp:97 mode-change-`

Parameter	Description
	`capability=2;mode-set=0,1,2,3,4,5,6,7,`
'SDP Ptime Answer' `sbc-sdp-ptime-ans` [IpProfile_ SBCSDPPtimeAnswer]	Defines the packetization time (ptime) of the coder in RTP packets for the SIP UA associated with the IP Profile. This is useful when implementing transrating. ■ [0] **Remote Answer** = (Default) Use ptime according to SDP answer. ■ [1] **Original Offer** = Use ptime according to SDP offer. ■ [2] **Preferred Value** = Use the ptime according to the 'Preferred Ptime' parameter (see below) if it is configured to a non-zero value. **Note:** ■ Regardless of the settings of this parameter, if a non-zero value is configured for the 'Preferred Ptime' parameter (see below), it is used as the ptime in the SDP offer.
'Preferred Ptime' `sbc-preferred-ptime` [IpProfile_ SBCPreferredPTime]	Defines the packetization time (ptime) in msec for the SIP UA associated with the IP Profile, in the outgoing SDP offer. If the 'SDP Ptime Answer' parameter (see above) is configured to Preferred Value [2] and the 'Preferred Ptime' parameter is configured to a non-zero value, the configured ptime is used (enabling ptime transrating if the other side uses a different ptime). If the 'SDP Ptime Answer' parameter is configured to Remote Answer [0] or Original Offer [1] and the 'Preferred Ptime' parameter is configured to a non-zero value, the configured value is used as the ptime in the SDP offer. The valid range is 0 to 200. The default is 0 (i.e., a preferred ptime is not used).
'Use Silence Suppression' `sbc-use-silence-supp` [IpProfile_ SBCUseSilenceSupp]	Defines silence suppression support for the SIP UA associated with the IP Profile ■ [0] **Transparent** = (Default) Forward as is. ■ [1] **Add** = Enable silence suppression for each relevant coder listed in the SDP. ■ [2] **Remove** = Disable silence suppression for each relevant coder listed in the SDP.

Parameter	Description
	Note: This feature requires DSP resources.
'RTP Redundancy Mode' `sbc-rtp-red-behav` [IpProfile_ SBCRTPRedundancyBehavior]	Enables interworking RTP redundancy negotiation support between SIP entities in the SDP offer-answer exchange (according to RFC 2198). The parameter defines the device's handling of RTP redundancy for the SIP UA associated with the IP Profile. According to the RTP redundancy SDP offer/answer negotiation, the device uses or discards the RTP redundancy packets. The parameter enables asymmetric RTP redundancy, whereby the device can transmit and receive RTP redundancy packets to and from a specific SIP UA, while transmitting and receiving regular RTP packets (no redundancy) for the other SIP UA involved in the voice path. The device can identify the RTP redundancy payload type in the SDP for indicating that the RTP packet stream includes redundant packets. RTP redundancy is indicated in SDP using the "red" coder type, for example: `a=rtpmap:<payload type> red/8000/1` RTP redundancy is useful when there is packet loss; the missing information may be reconstructed at the receiver side from the redundant packets. ■ [0] **As Is** = (Default) The device does not interfere in the RTP redundancy negotiation and forwards the SDP offer/answer (incoming and outgoing calls) as is without interfering in the RTP redundancy negotiation. ■ [1] **Enable** = The device always adds RTP redundancy capabilities in the outgoing SDP offer sent to the SIP UA. Whether RTP redundancy is implemented depends on the subsequent incoming SDP answer from the SIP UA. The device does not modify the incoming SDP offer received from the SIP UA, but if RTP redundancy is offered, it will support it in the outgoing SDP answer. Select the option if the SIP UA requires RTP redundancy. ■ [2] **Disable** = The device removes the RTP redundancy payload (if present) from the SDP offer/answer for calls received from or sent to the SIP UA. Select the option if the SIP UA does not support RTP redundancy. **Note:**

Parameter	Description
	■ To enable the device to generate RFC 2198 redundant packets, use the IPProfile_RTPRedundancyDepth parameter. ■ To configure the payload type in the SDP offer for RTP redundancy, use the RFC2198PayloadType.
'RTCP Mode' `sbc-rtcp-mode` [IPProfile_SBCRTCPMode]	Defines how the device handles RTCP packets during call sessions for the SIP UA associated with the IP Profile. This is useful for interworking RTCP between SIP entities. For example, this may be necessary when incoming RTCP is not compatible with the destination SIP UA's (this IP Profile) RTCP support. In such a scenario, the device can generate the RTCP and send it to the SIP UA. ■ [0] **Transparent** = (Default) RTCP is forwarded as is (unless transcoding is done, in which case, the device generates RTCP on both legs). ■ [1] **Generate Always** = Generates RTCP packets during active and inactive (e.g., during call hold) RTP periods (i.e., media is 'a=recvonly' or 'a=inactive' in the INVITE SDP). ■ [2] **Generate only if RTP Active** = Generates RTCP packets only during active RTP periods. In other words, the device does not generate RTCP when there is no RTP traffic (such as when a call is on hold). **Note:** The corresponding global parameter is [SBCRTCPMode].
'Jitter Compensation' `sbc-jitter-compensation` [IpProfile_ SBCJitterCompensation]	Enables the on-demand jitter buffer for SBC calls. The jitter buffer can be used when other functionality such as voice transcoding are not done on the call. The jitter buffer is useful when incoming packets are received at inconsistent intervals (i.e., packet delay variation). The jitter buffer stores the packets and sends them out at a constant rate (according to the coder's settings). ■ [0] **Disable** (default) ■ [1] **Enable** **Note:** ■ The jitter buffer parameters, 'Dynamic Jitter Buffer Minimum Delay' (DJBufMinDelay) and 'Dynamic Jitter

Parameter	Description
	Buffer Optimization Factor' (DJBufOptFactor) can be used to configure minimum packet delay only when transcoding is employed. ■ This feature may require DSP resources. For more information, contact the sales representative of your purchased device.
'ICE Mode' `ice-mode` [IPProfile_SBCIceMode]	Enables Interactive Connectivity Establishment (ICE) Lite for the SIP UA associated with the IP Profile. ICE is a methodology for NAT traversal, employing the Session Traversal Utilities for NAT (STUN) and Traversal Using Relays around NAT (TURN) protocols to provide a peer with a public IP address and port that can be used to connect to a remote peer. For example, ICE Lite is required when the device operates in Microsoft Teams Direct Routing (media bypass) environments. ■ [0] **Disable** (default) ■ [1] **Lite** For more information on ICE Lite, see ICE Lite. **Note:** As ICE has been defined by the WebRTC standard as mandatory, the support is important for deployments implementing WebRTC. For more information on WebRTC, see WebRTC.

'SDP Handle RTCP'
`sbc-sdp-handle-rtcp`
[IpProfile_ SBCSDPHandleRTCPAttribute]
	Enables the interworking of the RTCP attribute, 'a=rtcp' (RTCP) in the SDP, for the SIP UA associated with the IP Profile. The RTCP attribute is used to indicate the RTCP port for media when that port is not the next higher port number following the RTP port specified in the media line ('m=').
	The parameter is useful for SIP entities that either require the attribute or do not support the attribute. For example, Google Chrome and Web RTC do not accept calls without the RTCP attribute in the SDP. In Web RTC, Chrome (SDES) generates the SDP with 'a=rtcp', for example:
	`m=audio 49170 RTP/AVP 0`
	`a=rtcp:53020 IN IP6 2001:2345:6789:ABCD:EF01:2345:6789:ABCD`

Parameter	Description
	■ [0] **Don't Care** = (Default) The device forwards the SDP as is without interfering in the RTCP attribute (regardless if present or not). ■ [1] **Add** = The device adds the 'a=rtcp' attribute to the outgoing SDP offer sent to the SIP UA if the attribute was not present in the original incoming SDP offer. ■ [2] **Remove** = The device removes the 'a=rtcp' attribute, if present in the incoming SDP offer received from the other SIP UA, before sending the outgoing SDP offer to the SIP UA. **Note:** As the RTCP attribute has been defined by the WebRTC standard as mandatory, the support is important for deployments implementing WebRTC. For more information on WebRTC, see WebRTC.
'RTCP Mux' `sbc-rtcp-mux` [IPProfile_SBCRTCPMux]	Enables interworking of multiplexing of RTP and RTCP onto a single local port, between SIP entities. The parameter enables multiplexing of RTP and RTCP traffic onto a single local port, for the SIP UA associated with the IP Profile. Multiplexing of RTP data packets and RTCP packets onto a single local UDP port is done for each RTP session (according to RFC 5761). If multiplexing is not enabled, the device uses different (but adjacent) ports for RTP and RTCP packets. With the increased use of NAT and firewalls, maintaining multiple NAT bindings can be costly and also complicate firewall administration since multiple ports must be opened to allow RTP traffic. To reduce these costs and session setup times, support for multiplexing RTP data packets and RTCP packets onto a single port is advantageous. For multiplexing, the initial SDP offer must include the "a=rtcp-mux" attribute to request multiplexing of RTP and RTCP onto a single port. If the SDP answer wishes to multiplex RTP and RTCP, it must also include the "a=rtcp-mux" attribute. If the answer does not include the attribute, the offerer must not multiplex RTP and RTCP packets. If both ICE and multiplexed RTP-RTCP are used, the initial SDP offer must also include the "a=candidate:" attribute for both RTP and RTCP along with the "a=rtcp:"

Parameter	Description
	attribute, indicating a fallback port for RTCP in case the answerer does not support RTP and RTCP multiplexing. ■ [0] **Not Supported** = (Default) RTP and RTCP packets use different ports. ■ [1] **Supported** = Device multiplexes RTP and RTCP packets onto a single port. **Note:** As RTP multiplexing has been defined by the WebRTC standard as mandatory, the support is important for deployments implementing WebRTC. For more information on WebRTC, see WebRTC.
'RTCP Feedback' `sbc-rtcp-feedback` [IPProfile_SBCRTCPFeedback]	Enables RTCP-based feedback indication in outgoing SDPs sent to the SIP UA associated with the IP Profile. The parameter supports indication of RTCP-based feedback, according to RFC 5124, during RTP profile negotiation between two communicating SIP entities. RFC 5124 defines an RTP profile (S)AVPF for (secure) real-time communications to provide timely feedback from the receivers to a sender. For more information on RFC 5124, see http://tools.ietf.org/html/rfc5124. Some SIP entities may require RTP secure-profile feedback negotiation (AVPF/SAVPF) in the SDP offer/answer exchange, while other SIP entities may not support it. The device indicates whether or not feedback is supported on behalf of the SIP UA. It does this by adding an "F" or removing the "F" from the SDP media line ('m=') for AVP and SAVP. For example, the following shows "AVP" appended with an "F", indicating that the SIP UA is capable of receiving feedback `m=audio 49170 RTP/SAVPF 0 96` ■ [0] **Feedback Off** = (Default) The device does not send the feedback flag ("F") in SDP offers/answers that are sent to the SIP UA. If the SDP 'm=' attribute of an incoming message that is destined to the SIP UA includes the feedback flag, the device removes it before sending the message to the SIP UA. ■ [1] **Feedback On** = The device includes the feedback flag ("F") in the SDP offer sent to the SIP UA. The device includes the feedback flag in the SDP answer sent to the SIP UA only if it was present in the SDP

Parameter	Description
	offer received from the other SIP UA. ■ [2] **As Is** = The device does not involve itself in the feedback, but simply forwards any feedback indication as is. **Note:** ■ As RTCP-based feedback has been defined by the WebRTC standard as mandatory, the support is important for deployments implementing WebRTC. For more information on WebRTC, see WebRTC. ■ RTCP-based feedback is required for the VoIPerfect feature (see VoIPerfect).
'Re-number MID' `sbc-renumber-mid` [IpProfile_SBCRenumberMID]	Enables the device to change the value of the 'a=mid:n' attribute (where *n* is a unique value) in the outgoing SDP offer so that in the first media ('m=' line) the value will be 0, the next media the value will be 1, and so on. This is done only if the 'a=mid' attribute is present in the incoming SDP offer. ■ [0] **Disable** (default) ■ [1] **Enable** **Note:** ■ For deployments implementing WebRTC (see WebRTC), it's recommended that you configure the parameter to **Enable**.
'Voice Quality Enhancement' `sbc-voice-quality-enhancement` [IpProfile_ SBCVoiceQualityEnhancement]	Enables the device to detect speech and network quality (packet loss and bandwidth reduction) and triggers the device to overcome the adverse conditions to ensure high call quality. ■ [0] **Disable** (default) ■ [1] **Enable** **Note:** The parameter is applicable only to the VoIPerfect feature (see VoIPerfect).
'Max Opus Bandwidth' `sbc-max-opus-bandwidth` [IpProfile_SBCMaxOpusBW]	Defines the VoIPerfect mode of operation, which is based on the Opus coder. ■ **0** = (Default) Managed Opus ■ **80000** = Smart Transcoding

Parameter	Description
	Note: The parameter is applicable only to the VoIPerfect feature (see VoIPerfect).
'Generate No-Op Packets' `sbc-generate-noop` [IpProfile_ SBCGenerateNoOp]	Enables the device to send RTP or T.38 No-Op packets during RTP or T.38 silence periods. ■ [0] **Disable** (default) ■ [1] **Enable** For more information on No-Op packets, see No-Op Packets on page 143.
'SBC Multiple Coders' `configure voip > coders-and-profiles ip-profile > sbc-multiple-coders` [IpProfile_ SBCMultipleCoders]	Enables support of multiple coders in the SDP answer that is received from the peer side for the UA associated with this IP Profile. ■ [0] **Not Supported** = (Default) If multiple coders ('m=' line) are present in the SDP answer received from the peer side, the device uses only the first supported coder in the list for the RTP media. ■ [1] **Supported** = If multiple coders ('m=' line) are present in the SDP answer received from the peer side, the device does one of the following, depending on whether DSP resources are required (e.g., for DTMF transcoding): ✓ DSP resources required: Upon receipt of the SDP answer, the device sends a re-INVITE message with only a single coder (first supported coder in the list) to the UA associated with this IP Profile. In other words, the device “forces” the UAs to negotiate only a single coder for the RTP media. ✓ DSP resources not required: The device supports multiple coders in the SDP answer, allowing the RTP media to use any one of the listed coders (doesn’t send a re-INVITE).
'SBC Allow Only Negotiated PT' `configure voip > coders-and-profiles ip-profile > sbc-allow-only-negotiated-pt`	Enables the device to allow only media (RTP) packets, from the UA associated with this IP Profile, using the single coder (payload type) that was negotiated during the SDP offer/answer exchange (e.g., 'm=audio 53456 RTP/AVP 0' for G.711). The device drops all other packets from the UA using any other coder. ■ [0] **Disable** =(Default) The device allows packets with

Parameter	Description
[IpProfile_ SBCAllowOnlyNegotiatedPT]	multiple negotiated coders. ■ [1] **Enable** = The device allows only packets with the single negotiated coder.
Quality of Service	
'RTP IP DiffServ' `rtp-ip-diffserv` [IpProfile_IPDiffServ]	Defines the DiffServ value for Premium Media class of service (CoS) content. The valid range is 0 to 63. The default is 46. **Note:** The corresponding global parameter is [PremiumServiceClassMediaDiffServ].
'Signaling DiffServ' `signaling-diffserv` [IpProfile_SigIPDiffServ]	Defines the DiffServ value for Premium Control CoS content (Call Control applications). The valid range is 0 to 63. The default is 40. **Note:** The corresponding global parameter is [PremiumServiceClassControlDiffServ].
'Data DiffServ' `data-diffserv` [IpProfile_DataDiffServ]	Defines the DiffServ value of MSRP traffic in the IP header's DSCP field. The valid range is 0 to 63. The default is 0. For more information on MSRP, see Configuring Message Session Relay Protocol on page 779.
Jitter Buffer	
'Dynamic Jitter Buffer Minimum Delay' `jitter-buffer-minimum-delay` [IpProfile_JitterBufMinDelay]	Defines the minimum delay (in msec) of the device's dynamic Jitter Buffer. The valid range is 0 to 150. The default delay is 10. For more information on Jitter Buffer, see Configuring the Dynamic Jitter Buffer. **Note:** The corresponding global parameter is DJBufMinDelay.
'Dynamic Jitter Buffer Optimization Factor' `jitter-buffer-optimization-factor` [IpProfile_JitterBufOptFactor]	Defines the Dynamic Jitter Buffer frame error/delay optimization factor. The valid range is 0 to 12. The default factor is 10. For more information on Jitter Buffer, see Configuring the Dynamic Jitter Buffer. **Note:** ■ For data (fax and modem) calls, set the parameter to 12.

Parameter	Description
	■ The corresponding global parameter is DJBufOptFactor.
'Jitter Buffer Max Delay' `jitter-buffer-max-delay` [IpProfile_JitterBufMaxDelay]	Defines the maximum delay and length (in msec) of the Jitter Buffer. The valid range is 150 to 2,000. The default is 250.
Voice	
'Echo Canceler' `echo-canceller` [IpProfile_ EnableEchoCanceller]	Enables the device's Echo Cancellation feature (i.e., echo from voice calls is removed). ■ [0] **Disable** ■ [1] **Line** (default) ■ [2] **Acoustic** For a detailed description of the Echo Cancellation feature, see Configuring Echo Cancellation. **Note:** The corresponding global parameter is EnableEchoCanceller.
'Input Gain' `input-gain` [IpProfile_InputGain]	Defines the pulse-code modulation (PCM) input gain control (in decibels). The valid range is -32 to 31 dB. The default is 0 dB. **Note:** The corresponding global parameter is InputGain.
'Voice Volume' `voice-volume` [IpProfile_VoiceVolume]	Defines the voice gain control (in decibels). The valid range is -32 to 31 dB. The default is 0 dB. **Note:** The corresponding global parameter is VoiceVolume.
SBC Signaling	
'PRACK Mode' `sbc-prack-mode` [IpProfile_SbcPrackMode]	Defines the device's handling of SIP PRACK messages for the SIP UA associated with the IP Profile. ■ [0] **Disabled** = The device doesn't allow PRACK: ✓ For SIP requests (INVITE) and responses (18x), the device removes the '100rel' option from the SIP Supported header (if present). In other words, the device disables PRACK with this SIP UA.

Parameter	Description
	✓ If the device receives an INVITE message containing the header and value 'Require: 100rel', it rejects the message (with a SIP 420 response). ✓ If the device receives a SIP 18x response containing the RSeq header and the '100rel' option, it sends a CANCEL message to cancel the SIP dialog. ■ [1] **Optional** = PRACK is optional. If required, the device performs the PRACK process on behalf of the SIP UA. ■ [2] **Mandatory** = PRACK is required for this SIP UA. Calls from UAs that do not support PRACK are rejected. Calls destined to these UAs are also required to support PRACK. ■ [3] **Transparent** = (Default) The device does not intervene with the PRACK process and forwards the request as is. ■ [4] **Optional With Adaptations** = This option may be useful, for example, to prevent PRACK congestion caused by the flooding of the device with 18x messages without body. ✓ Outgoing INVITE messages (sent to SIP UA): - The device adds the header 'Supported: 100rel' to the INVITE message. If the message included the header and value 'Require: 100rel', it removes the '100rel' option. - If the device adds the '100rel' option, it terminates and fully handles PRACK; otherwise, the device forwards the message transparently. ✓ Incoming INVITE messages (from SIP UA): - If the message doesn't contain the '100rel' option, the device doesn't handle PRACK. - If the message contains the header and value 'Require: 100rel', the device processes PRACK as described for the **Mandatory** optional value above (and terminates PRACK, if necessary). - If the message contains the header and value 'Supported: 100rel', the device activates PRACK Extensions as follows:

Parameter	Description
	>> If the device sends an outgoing 18x responses with body (e.g., SDP), the device processes PRACK as described for the **Mandatory** optional value above (and terminates PRACK, if necessary). >> If the device sends an outgoing 18x responses without body, the device removes the '100rel' option and the RSeq header (if present). If the RSeq header was present, the device sends a terminated PRACK to the incoming leg without Optional With Adaptations outgoing leg involvement.
'P-Asserted-Identity Header Mode' `sbc-assert-identity` [IpProfile_SBCAssertIdentity]	Defines the device's handling of the SIP P-Asserted-Identity header for the SIP UA associated with the IP Profile. This header indicates how the outgoing SIP message asserts identity. ■ [0] **As Is** = (Default) P-Asserted Identity header is not affected and the device uses the same P-Asserted-Identity header (if present) in the incoming message for the outgoing message. ■ [1] **Add** = Adds a P-Asserted-Identity header. The header's values are taken from the source URL. ■ [2] **Remove** = Removes the P-Asserted-Identity header. **Note:** ■ The parameter affects only initial INVITE requests. ■ The corresponding global parameter is [SBCAssertIdentity].
'Diversion Header Mode' `sbc-diversion-mode` [IpProfile_ SBCDiversionMode]	Defines the device's handling of the SIP Diversion header for the SIP UA associated with the IP Profile. ■ [0] **As Is** = (Default) Diversion header is not handled. ■ [1] **Add** = History-Info header is converted to a Diversion header. ■ [2] **Remove** = Removes the Diversion header and the conversion to the History-Info header depends on the SBCHistoryInfoMode parameter. For more information on interworking of the History-Info and Diversion headers, see Interworking SIP Diversion and

Parameter	Description
	History-Info Headers. **Note:** ■ If the Diversion header is used, you can specify the URI type (e.g., "tel:") to use in the header, using the [SBCDiversionUriType] parameter.
'History-Info Header Mode' `sbc-history-info-mode` [IpProfile_ SBCHistoryInfoMode]	Defines the device's handling of the SIP History-Info header for the SIP UA associated with the IP Profile. ■ [0] **As Is** = (Default) History-Info header is not handled. ■ [1] **Add** = Diversion header is converted to a History-Info header. ■ [2] **Remove** = History-Info header is removed from the SIP dialog and the conversion to the Diversion header depends on the SBCDiversionMode parameter. For more information on interworking of the History-Info and Diversion headers, see Interworking SIP Diversion and History-Info Headers.
'Session Expires Mode' `sbc-session-expires-mode` [IpProfile_ SBCSessionExpiresMode]	Defines the required session expires mode for the SIP UA associated with the IP Profile. ■ [0] **Transparent** = (Default) The device does not interfere with the session expires negotiation. ■ [1] **Observer** = If the SIP Session-Expires header is present, the device does not interfere, but maintains an independent timer for each leg to monitor the session. If the session is not refreshed on time (participant's time plus a grace period), the device disconnects the call. The grace period is 120 seconds on the client (incoming) side and 60 seconds on the server (outgoing) side. ■ [2] **Not Supported** = The device does not allow a session timer with this SIP UA. ■ [3] **Supported** = The device enables the session timer with this SIP UA. If the incoming SIP message does not include any session timers, the device adds the session timer information to the sent message. You can configure the value of the Session-Expires and Min-SE headers, using the [SBCSessionExpires] and [SBCMinSE] parameters, respectively.

Parameter	Description
'SIP UPDATE Support' sbc-rmt-update-supp [IpProfile_ SBCRemoteUpdateSupport]	Defines if the SIP UA associated with this IP Profile supports the receipt of SIP UPDATE messages. ■ [0] **Not Supported** = The UA doesn't support the receipt of UPDATE messages. ■ [1] **Supported Only After Connect** = The UA supports the receipt of UPDATE messages, but only after the call is connected. ■ [2] **Supported** = (Default) The UA supports the receipt of UPDATE messages during call setup and after call establishment. ■ [3] **According Remote Allow** = For refreshing the timer of currently active SIP sessions, the device sends session refreshes using SIP UPDATE messages only if the SIP Allow header in the last SIP message received from the user contains the value "UPDATE". If the Allow header does not contain the "UPDATE" value (or if the parameter is not configured to this option), the device uses INVITE messages for session refreshes.
'Remote re-INVITE' sbc-rmt-re-invite-supp [IpProfile_ SBCRemoteReinviteSupport]	Defines if the SIP UA associated with this IP Profile supports the receipt of SIP re-INVITE messages. ■ [0] **Not Supported** = The UA doesn't support the receipt of re-INVITE messages. If the device receives a re-INVITE from another UA that is destined to this UA, the device "terminates" the re-INVITE and sends a SIP response to the UA that sent it, which can be a success or a failure, depending on whether the device can bridge the media between the UAs. ■ [1] **Supported only with SDP** = The UA supports the receipt of re-INVITE messages, but only if they contain an SDP body. If the incoming re-INVITE from another UA doesn't contain SDP, the device creates and adds an SDP body to the re-INVITE that it forwards to the UA. ■ [2] **Supported** = (Default) The UA supports the receipt of re-INVITE messages with or without SDP.
'Remote Delayed Offer Support' sbc-rmt-delayed-	Defines if the remote UA supports delayed offer (i.e., initial INVITE requests without an SDP offer). ■ [0] **Not Supported**

Parameter	Description
offer [IpProfile_SBCRemoteDelayedOfferSupport]	■ [1] **Supported** (default) **Note:** ■ For the parameter to function, you need to assign extension coders to the IP Profile of the SIP UA that does not support delayed offer (using the IpProfile_SBCExtensionCodersGroupName parameter).
'MSRP re-INVITE/UPDATE' sbc-msrp-re-invite-update-supp [IpProfile_SBCMSRPReinviteUpdateSupport]	Defines if the SIP UA (MSRP endpoint) associated with this IP Profile supports the receipt of re-INVITE and UPDATE SIP messages. ■ [0] **Not Supported** = The device doesn't send re-INVITE or UPDATE messages to the UA. If the device receives any of these messages from the peer UA, the device "terminates" the messages, and then sends a SIP response to the peer UA on behalf of the UA associated with this IP Profile. ■ [1] **Supported** (default) For more information on MSRP, see Configuring Message Session Relay Protocol on page 779.
'MSRP Offer Setup Role' sbc-msrp-offer-setup-role [IpProfile_SBCMSRPOfferSetupRole]	Defines the device's preferred MSRP role, which is indicated in the initial SDP offer that it sends to the destination MSRP endpoint ('a=setup' line) associated with this IP Profile. However, this is only a preferred role; the actual role that the device takes on depends on the destination MSRP endpoint's desired role, which is indicated in the SDP answer in its reply to the device: ■ If 'a=setup:active', the device takes the passive role. ■ If 'a=setup:passive', the device takes the active role. ■ If 'a=setup' (i.e., empty) or no 'a=setup', the device takes the active role. The possible values include: ■ [0] **Active** = The device prefers the active role and includes 'a=setup:active' in the outgoing SDP offer sent to the endpoint associated with the IP Profile. ■ [1] **Passive** = The device prefers the passive role and includes 'a=setup:passive' in the outgoing SDP offer sent to the endpoint associated with the IP Profile.

Parameter	Description
	■ [2] **ActPass** = (Default) The device has no role preference and includes 'a=setup:actpass' in the outgoing SDP offer sent to the endpoint associated with the IP Profile For more information on MSRP, see Configuring Message Session Relay Protocol on page 779.
'MSRP Empty Message Format' `sbc-msrp-empty-message-format` [IPProfile_SBCMSRPEmpMsg]	On an active MSRP leg, enables the device to add the Content-Type header to the first empty (i.e., no body) MSRP message that is used to initiate the MSRP connection. ■ [0] **Default** = (Default) Sends the empty message with regular headers, according to the RFC for MSRP. ■ [1] **With Content Type** = Adds the Content-Type header to the empty message (in addition to the regular headers according to the RFC for MSRP). For more information on MSRP, see Configuring Message Session Relay Protocol on page 779.
'Remote Representation Mode' `sbc-rmt-rprsntation` [IpProfile_ SBCRemoteRepresentationMode]	Enables interworking SIP in-dialog, Contact and Record-Route headers between SIP entities. The parameter defines the device's handling of in-dialog, Contact and Record-Route headers for messages sent to the SIP UA associated with the IP Profile. ■ [-1] **According to Operation Mode** = (Default) Depends on the setting of the 'Operation Mode' parameter in the IP Groups or SRDs table: ✓ B2BUA: Device operates as if the parameter is set to Replace Contact [0]. ✓ Call State-full Proxy: Device operates as if the parameter is set to Add Routing Headers [1]. ■ [0] **Replace Contact** = The URI host part in the Contact header of the received message (from the other side) is replaced with the device's address or with the value of the 'SIP Group Name' parameter (configured in the IP Groups table) in the outgoing message sent to the SIP UA. ■ [1] **Add Routing Headers** = Device adds a Record-Route header for itself to outgoing messages

Parameter	Description
	(requests\responses) sent to the SIP UA in dialog-setup transactions. The Contact header remains unchanged. ■ [2] **Transparent** = Device doesn't change the Contact header and doesn't add a Record-Route header for itself. Instead, it relies on its' own inherent mechanism to remain in the route of future requests in the dialog (for example, relying on the way the endpoints are set up or on TLS as the transport type).
'Keep Incoming Via Headers' `sbc-keep-via-headers` [IpProfile_ SBCKeepVIAHeaders]	Enables interworking SIP Via headers between SIP entities. The parameter defines the device's handling of Via headers for messages sent to the SIP UA associated with the IP Profile. ■ [-1] **According to Operation Mode** = Depends on the setting of the 'Operation Mode' parameter in the IP Groups table or SRDs table: ✓ B2BUA: Device operates as if the parameter is set to Disable [0]. ✓ Call State-full Proxy: Device operates as if the parameter is set to Enable [1]. ■ [0] **Disable** = Device removes all Via headers received in the incoming SIP request from the other leg and adds a Via header identifying only itself, in the outgoing message sent to the SIP UA. ■ [1] **Enable** = Device retains the Via headers received in the incoming SIP request and adds itself as the top-most listed Via header in the outgoing message sent to the SIP UA.
'Keep Incoming Routing Headers' `sbc-keep-routing-headers` [IpProfile_ SBCKeepRoutingHeaders]	Enables interworking SIP Record-Route headers between SIP entities. The parameter defines the device's handling of Record-Route headers for request/response messages sent to the the SIP UA associated with the IP Profile. ■ [-1] **According to Operation Mode** = (Default) Depends on the setting of the 'Operation Mode' in the IP Group or SRDs table: ✓ B2BUA: Device operates as if the parameter is set to Disable [0].

Parameter	Description
	✓ Call State-full Proxy: Device operates as if the parameter is set to Enable [1]. ■ [0] **Disable** = Device removes the Record-Route headers received in requests and responses from the other side, in the outgoing SIP message sent to the SIP UA. The device creates a route set for that side of the dialog based on these headers, but doesn't send them to the SIP UA. ■ [1] **Enable** = Device retains the incoming Record-Route headers received in requests and non-failure responses from the other side, in the following scenarios: ✓ The message is part of a SIP dialog-setup transaction. ✓ The messages in the setup and previous transaction didn't include the Record-Route header, and therefore hadn't set the route set. **Note:** Record-Routes are kept only for SIP INVITE, UPDATE, SUBSCRIBE and REFER messages.
'Keep User-Agent Header' `sbc-keep-user-agent` [IpProfile_ SBCKeepUserAgentHeader]	Enables interworking SIP User-Agent headers between SIP entities. The parameter defines the device's handling of User-Agent headers for response/request messages sent to the SIP UA associated with the IP Profile. ■ [-1] **According to Operation Mode** = (Default) Depends on the setting of the 'Operation Mode' parameter in the IP Group or SRDs table: ✓ B2BUA: Device operates as if this parameter is set to Disable [0]. ✓ Call State-full Proxy: Device operates as if this parameter is set to Enable [1]. ■ [0] **Disable** = Device removes the User-Agent/Server headers received in the incoming message from the other side, and adds its' own User-Agent header in the outgoing message sent to the SIP UA. ■ [1] **Enable** = Device retains the User-Agent/Server headers received in the incoming message and sends the headers as is in the outgoing message to the SIP UA.

Parameter	Description
'Handle X-Detect' sbc-handle-xdetect [IpProfile_SBCHandleXDetect]	Enables the detection and notification of events (AMD, CPT, and fax), using the X-Detect SIP header. ■ [0] **No** (default) ■ [1] **Yes** For more information, see Event Detection and Notification using X-Detect Header.
'ISUP Body Handling' sbc-isup-body-handling [IpProfile_SBCISUPBodyHandling]	Defines the handling of ISUP data for interworking SIP and SIP-I endpoints. ■ [0] **Transparent** = (Default) ISUP data is passed transparently (as is) between endpoints (SIP-I to SIP-I calls). ■ [1] **Remove** = ISUP body is removed from INVITE messages. ■ [2] **Create** = ISUP body is added to outgoing INVITE messages. ■ [3] **Create If Not Exists** = ISUP body is added to outgoing INVITE messages if it does not exist in the incoming leg. If it exists, unknown fields and messages by the device are passed transparently, while known fields can be manipulated using Message Manipulation rules. For known fields, some values that are "reserved for national use" may be changed to default. For more information on interworking SIP and SIP-I, see Interworking SIP and SIP-I Endpoints.
'ISUP Variant' sbc-isup-variant [IpProfile_SBCISUPVariant]	Defines the ISUP variant for interworking SIP and SIP-I endpoints. ■ [0] **itu92** = (Default) ITU 92 variant ■ [1] **Spirou** = SPIROU (ISUP France)
'Max Call Duration' sbc-max-call-duration [IpProfile_SBCMaxCallDuration]	Defines the maximum duration (in minutes) per SBC call that is associated with the IP Profile. If the duration is reached, the device terminates the call. The valid range is 0 to 35,791, where 0 is unlimited duration. The default is the value configured for the global parameter [SBCMaxCallDuration].
SBC Registration	

Parameter	Description
'User Registration Time' sbc-usr-reg-time [IpProfile_ SBCUserRegistrationTime]	Defines the registration time (in seconds) that the device responds to SIP REGISTER requests from users belonging to the SIP UA associated with the IP Profile. The registration time is inserted in the Expires header in the outgoing response sent to the user. The Expires header determines the lifespan of the registration. For example, a value of 3600 means that the registration will timeout in one hour and at that point, the user will not be able to make or receive calls. The valid range is 0 to 2,000,000. The default is 0. If configured to 0, the Expires header's value received in the user's REGISTER request remains unchanged. If no Expires header is received in the REGISTER message and the parameter is set to 0, the Expires header's value is set to 180 seconds, by default. **Note:** The corresponding global parameter is SBCUserRegistrationTime.
'NAT UDP Registration Time' sbc-usr-udp-nat-reg-time [IpProfile_ SBCUserBehindUdpNATRegistrationTime]	Defines the registration time (in seconds) that the device includes in register responses, in response to SIP REGISTER requests from users belonging to the SIP UA associated with the IP Profile. The parameter applies only to users that are located behind NAT and whose communication type is UDP. The registration time is inserted in the Expires header in the outgoing response sent to the user. The Expires header determines the lifespan of the registration. For example, a value of 3600 means that the registration will timeout in one hour, unless the user sends a refresh REGISTER before the timeout. Upon timeout, the device removes the user's details from the registration database, and the user will not be able to make or receive calls through the device. The valid value is 0 to 2,000,000. If configured to 0, the Expires header's value received in the user's REGISTER request remains unchanged. By default, no value is defined (-1). **Note:** If the parameter is not configured, the registration time is according to the global parameter SBCUserRegistrationTime or IP Profile parameter IpProfile_SBCUserRegistrationTime.

Parameter	Description
'NAT TCP Registration Time' sbc-usr-tcp-nat-reg-time [IpProfile_SBCUserBehindTcpNATRegistrationTime]	Defines the registration time (in seconds) that the device includes in register responses, in response to SIP REGISTER requests from users belonging to the SIP UA associated with the IP Profile. The parameter applies only to users that are located behind NAT and whose communication type is TCP. The registration time is inserted in the Expires header in the outgoing response sent to the user. The Expires header determines the lifespan of the registration. For example, a value of 3600 means that the registration will timeout in one hour, unless the user sends a refresh REGISTER before the timeout. Upon timeout, the device removes the user's details from the registration database, and the user will not be able to make or receive calls through the device. The valid value is 0 to 2,000,000. If configured to 0, the Expires header's value received in the user's REGISTER request remains unchanged. By default, no value is defined (-1). **Note:** If the parameter is not configured, the registration time is according to the global parameter SBCUserRegistrationTime or IP Profile parameter IpProfile_SBCUserRegistrationTime.
SBC Forward and Transfer	
'Remote REFER Mode' sbc-rmt-refer-behavior [IpProfile_SBCRemoteReferBehavior]	Defines the device's handling of SIP REFER requests for the SIP UA (transferee - call party that is transfered to the transfer target) associated with the IP Profile. ■ [0] **Regular** = (Default) SIP Refer-To header value is unchanged and the device forwards the REFER message as is. However, if you configure the 'Remote Replaces Mode' parameter (see below) to any value other than **Keep as is**, the device may modify the URI of the Refer-To header to reflect the call identifiers of the leg. ■ [1] **Database URL** = SIP Refer-To header value is changed so that the re-routed INVITE is sent through the device: a. Before forwarding the REFER request, the device changes the host part to the device's IP address

Parameter	Description
	and adds a special prefix ("T~&R_") to the Contact user part. **b.** The incoming INVITE is identified as a REFER-resultant INVITE according to this special prefix. **c.** The device replaces the host part in the Request-URI with the host from the REFER contact. The special prefix remains in the user part for regular classification, manipulation, and routing. The special prefix can also be used for specific routing rules for REFER-resultant INVITEs. **d.** The special prefix is removed before the resultant INVITE is sent to the destination ((transfer target). ■ [2] **IP Group Name** = Changes the host part in the REFER message to the value that you configured for the 'SIP Group Name' parameter in the IP Groups table (see Configuring IP Groups on page 451). ■ [3] **Handle Locally** = Handles the incoming REFER request itself without forwarding the REFER. The device generates a new INVITE to the alternative destination (transfer target) according to the rules in the IP-to-IP Routing table (the 'Call Trigger' parameter must be set to REFER). ■ [4] **Local Host** = In the REFER message received from the transferor, the device replaces the Refer-To header value (URL) with the IP address of the device or with the 'Local Host Name' parameter value configured for the IP Group (transferee) to where the device forwards the REFER message. This ensures that the transferee sends the re-routed INVITE back to the device which then sends the call to the transfer target. ■ [5] **Keep URI (user@host)** = The device forwards the REFER message without changing the URI (user@host) in the SIP Refer-To header. If you configure the 'Remote Replaces Mode' parameter (see below) to any value other than **Keep as is**, the devicemay modify the 'replaces' parameter of the Refer-To header to reflect the call identifiers of the leg. This applies to all types of call transfers (e.g., blind and attendant transfer).

Parameter	Description
	Note: ■ You can override the parameter's settings using Message Manipulation rules configured with the AudioCodes proprietary SIP header, X-AC-Action. For more information, see Using the Proprietary SIP X-AC-Action Header on page 772. ■ The corresponding global parameter is [SBCReferBehavior]. ■ For MSRP sessions, the **Handle Locally** option is not applicable. For more information on MSRP sessions, see Configuring Message Session Relay Protocol on page 779.
'Remote Replaces Mode' `sbc-rmt-replaces-behavior` [IpProfile_SBCRemoteReplacesBehavior]	Enables the device to handle incoming INVITEs containing the Replaces header for the SIP UA (which does not support the header) associated with the IP Profile. The Replaces header is used to replace an existing SIP dialog with a new dialog such as in call transfer or call pickup. ■ [0] **Standard** = (Default) The SIP UA supports INVITE messages containing Replaces headers. The device forwards the INVITE message containing the Replaces header to the SIP UA. The device may change the value of the Replaces header to reflect the call identifiers of the leg. ■ [1] **Handle Locally** = The SIP UA does not support INVITE messages containing Replaces headers. The device terminates the received INVITE containing the Replaces header and establishes a new call between the SIP UA and the new call party. It then disconnects the call with the initial call party, by sending it a SIP BYE request. ■ [2] **Keep as is** = The SIP UA supports INVITE messages containing Replaces headers. The device forwards the Replaces header as is in incoming REFER and outgoing INVITE messages from/to the SIP UA (i.e., Replaces header's value is unchanged). For example, assume that the device establishes a call between A and B. If B initiates a call transfer to C, the device receives an INVITE with the Replaces header from

Parameter	Description
	C. If A supports the Replaces header, the device simply forwards the INVITE as is to A; a new call is established between A and C and the call between A and B is disconnected. However, if A does not support the Replaces header, the device uses this feature to terminate the INVITE with Replaces header and handles the transfer for A. The device does this by connecting A to C, and disconnecting the call between A and B, by sending a SIP BYE request to B. Note that if media transcoding is required, the device sends an INVITE to C on behalf of A with a new SDP offer.
'Play RBT To Transferee' `sbc-play-rbt-to-xferee` [IpProfile_SBCPlayRBTToTransferee]	Enables the device to play a ringback tone to the transferred party (transferee) during a blind call transfer, for the SIP UA associated with the IP Profile (which does not support such a tone generation during call transfer). The ringback tone indicates to the transferee of the ringing of the transfer target (to where the transferee is being transferred). ■ [0] **No** (Default) ■ [1] **Yes** Typically, the transferee hears a ringback tone only if the transfer target sends it early media. However, if the transferee is put on-hold before being transferred, no ringback tone is heard. When this feature is enabled, the device generates a ringback tone to the transferee during call transfer in the following scenarios: ■ Transfer target sends a SIP 180 (Ringing) to the device. ■ For non-blind transfer, if the call is transferred while the transfer target is ringing and no early media occurs. ■ The 'Remote Early Media RTP Behavior parameter is set to Delayed (used in the Skype for Business environment), and transfer target sends a 183 Session Progress with SDP offer. If early media from the transfer target has already been detected, the transferee receives RTP stream from the transfer target. If it has not been detected, the device generates a ringback tone to the transferee and stops

Parameter	Description
	the tone generation once RTP has been detected from the transfer target. For any of these scenarios, if the transferee is put on-hold by the transferor, the device retrieves the transferee from hold, sends a re-INVITE if necessary, and then plays the ringback tone. **Note:** For the device to play the ringback tone, it must be loaded with a Prerecorded Tones (PRT) file. For more information, see Prerecorded Tones File.
'Remote 3xx Mode' `sbc-rmt-3xx-behavior` [IpProfile_ SBCRemote3xxBehavior]	Defines the device's handling of SIP 3xx redirect responses for the SIP UA associated with the IP Profile. By default, the device's handling of SIP 3xx responses is to send the Contact header unchanged. However, some SIP entities may support different versions of the SIP 3xx standard while others may not even support SIP 3xx. When enabled, the device handles SIP redirections between different subnets (e.g., between LAN and WAN sides). This is required when the new address provided by the redirector (Redirect sever) may not be reachable by the far-end user (FEU) located in another subnet. For example, a far-end user (FEU) in the WAN sends a SIP request via the device to a Redirect server in the LAN, and the Redirect server replies with a SIP 3xx response to a PBX in the LAN in the Contact header. If the device sends this response as is (i.e., with the original Contact header), the FEU is unable to reach the new destination. ■ [0] **Transparent** = (Default) The device forwards the received SIP 3xx response as is, without changing the Contact header (i.e., transparent handling). ■ [1] **Database URL** = The device changes the Contact header so that the re-route request is sent through the device. The device changes the URI in the Contact header of the received SIP 3xx response to its own URI and adds a special user prefix ("T~&R_"), which is then sent to the FEU. The FEU then sends a new INVITE to the device, which the device then sends to the correct destination. ■ [2] **Handle Locally** = The device handles SIP 3xx responses on behalf of the dialog-initiating UA and retries the request (e.g., INVITE) using one or more

Parameter	Description
	alternative URIs included in the 3xx response. The device sends the new request to the alternative destination according to the IP-to-IP Routing table (the 'Call Trigger' field must be set to **3xx**). ■ [3] **IP Group Name** = If the 'SIP Group Name' parameter of the IP Group of the dialog-initiating UA is configured with a non-empty value, the device changes the host part of the Contact header in the 3xx response to this value, before forwarding the 3xx response to the dialog-initiating UA. ■ [4] **Local Host** = The device changes the host part of the Contact header in the 3xx response before forwarding the 3xx response to the dialog-initiating UA. If the 'Local Host Name' parameter of the IP Group of the dialog-initiating UA is configured with a non-empty value, the device changes the host part of the Contact header to this value. If the 'Local Host Name' is empty, the device changes the host part to the device's IP address (the same IP address used in the SIP Via and Contact headers of messages sent to the IP Group). **Note:** ■ When the parameter is changed from **Database URL** to **Transparent**, new 3xx Contact headers remain unchanged. However, requests with the special prefix continue using the device's database to locate the new destination. ■ Optional values **IP Group Name** and **Local Host** are applicable only to 3xx responses received due to INVITE messages. ■ Only one database entry is supported for the same host, port, and transport combination. For example, the following URLs cannot be distinguished by the device: ✓ sip:10.10.10.10:5060;transport=tcp;param=a ✓ sip:10.10.10.10:5060;transport=tcp;param=b ■ The database entry expires two hours after the last use.

Parameter	Description
	■ The maximum number of destinations (i.e., database entries) is 50. ■ The corresponding global parameter is SBC3xxBehavior.
SBC Hold	
'Remote Hold Format' remote-hold-Format [IPProfile_ SBCRemoteHoldFormat]	Defines the format of the SDP in the SIP re-INVITE (or UPDATE) for call hold that the device sends to the held party. ■ [0] **Transparent** = (Default) Device forwards SDP as is. ■ [1] **Send Only** = Device sends SDP with 'a=sendonly'. ■ [2] **Send Only Zero Ip** = Device sends SDP with 'a=sendonly' and 'c=0.0.0.0'. ■ [3] **Inactive** = Device sends SDP with 'a=inactive'. ■ [4] **Inactive Zero Ip** = Device sends SDP with 'a=inactive' and 'c=0.0.0.0'. ■ [5] **Not Supported** = This option can be used when the remote side does not support call hold. The device terminates call hold requests received on the leg interfacing with the initiator of the call hold, and replies to this initiator with a SIP 200 OK response. However, call retrieve (resume) requests received from the initiator are forwarded to the remote side. The device can play a held tone to the held party if the 'Play Held Tone' parameter is set to **Internal**. ■ [6] **Hold and Retrieve Not Supported** = This option can be used when the remote side does not support call hold and retrieve (resume). The device terminates call hold and call retrieve requests received on the leg interfacing with the initiator of the call hold/retrieve, and replies to this initiator with a SIP 200 OK response. Therefore, the device does not forward call hold and/or retrieve requests to the remote side.
'Reliable Held Tone Source' reliable-heldtone-source [IPProfile_ ReliableHoldToneSource]	Enables the device to consider the received call-hold request (re-INVITE/UPDATE) with SDP containing 'a=sendonly', as genuine. ■ [0] **No** = (Default) Even if the received SDP contains

Parameter	Description
	'a=sendonly', the device plays a held tone to the held party. This is useful in cases where the initiator of the call hold does not support the generation of held tones. ■ [1] **Yes** = If the received SDP contains 'a=sendonly', the device does not play a held tone to the held party (and assumes that the initiator of the call hold plays the held tone). **Note:** The device plays a held tone only if the 'Play Held Tone' parameter is set to **Internal** or **External**.
'Play Held Tone' `play-held-tone` [IpProfile_SBCPlayHeldTone]	Enables the device to play Music-on-Hold (MoH) to call parties that are placed on hold. This is useful if the held party does not support the play of a local hold tone, or for IP entities initiating call hold that do not support the generation of hold tones. ■ [0] No = (Default) The device does not play any tone to held call parties. ■ [1] Internal = Plays the local default hold tone or a tone defined in the PRT file (if installed). ■ [2] External = Plays MoH audio streams that originate from an external media source. For more information, see Configuring SBC MoH from External Media Source on page 791 **Note:** If you configure the parameter to **Internal**, the device plays the tone only if the 'SBC Remote Hold Format' parameter is configured to one of the following: send-only, send only 0.0.0.0, not supported, or transparent (when the incoming SDP is 'sendonly').
SBC Fax	
'Fax Coders Group' `sbc-fax-coders-group-name` [IpProfile_ SBCFaxCodersGroupName]	Assigns a Coder Group which defines the supported fax coders for fax negotiation for the SIP UA associated with the IP Profile. To configure Coder Groups, see Configuring Coder Groups. **Note:** The parameter is applicable only if you set the IpProfile_SBCFaxBehavior parameter to a value other than [0].
'Fax Mode'	Enables the device to handle fax offer-answer

Parameter	Description
`sbc-fax-behavior` [IpProfile_SBCFaxBehavior]	negotiations for the SIP UA associated with the IP Profile. ■ [0] **As Is** = (Default) Device forwards fax transparently, without interference. ■ [1] **Handle always** = Handle fax according to fax settings in the IP Profile for all offer-answer transactions (including the initial INVITE). ■ [2] **Handle on re-INVITE** = Handle fax according to fax settings in the IP Profile for all re-INVITE offer-answer transactions (except for initial INVITE). **Note:** The fax settings in the IP Profile include IpProfile_SBCFaxCodersGroupName, IpProfile_SBCFaxOfferMode, and IpProfile_SBCFaxAnswerMode.
'Fax Offer Mode' `sbc-fax-offer-mode` [IpProfile_SBCFaxOfferMode]	Defines the coders included in the outgoing SDP offer (sent to the called "fax") for the SIP UA associated with the IP Profile. ■ [0] **All coders** = (Default) Use only (and all) the coders of the selected Coder Group configured using the SBCFaxCodersGroupID parameter. ■ [1] **Single coder** = Use only one coder. If a coder in the incoming offer (from the calling "fax") matches a coder in the SBCFaxCodersGroupID, the device uses this coder. If no match exists, the device uses the first coder listed in the Coders Group ID (SBCFaxCodersGroupID). **Note:** The parameter is applicable only if you set the IpProfile_SBCFaxBehavior parameter to a value other than [0].
'Fax Answer Mode' `sbc-fax-answer-mode` [IpProfile_ SBCFaxAnswerMode]	Defines the coders included in the outgoing SDP answer (sent to the calling "fax") for the SIP UA associated with the IP Profile. ■ [0] **All coders** = Use matched coders between the incoming offer coders (from the calling "fax") and the coders of the selected Coder Group (configured using the SBCFaxCodersGroupID parameter). ■ [1] **Single coder** = (Default) Use only one coder. If the incoming answer (from the called "fax") includes a coder that matches a coder match between the

Parameter	Description
	incoming offer coders (from the calling "fax") and the coders of the selected Coder Group (SBCFaxCodersGroupID, then the device uses this coder. If no match exists, the device uses the first listed coder of the matched coders between the incoming offer coders (from the calling "fax") and the coders of the selected Coder Group. **Note:** The parameter is applicable only if you set the IpProfile_SBCFaxBehavior parameter to a value other than [0].
'Remote Renegotiate on Fax Detection' `sbc-rmt-renegotiate-on-fax-detect` [IPProfile_SBCRemoteRenegotiateOnFaxDetection]	Enables local handling of fax detection and negotiation by the device on behalf of the SIP UA associated with the IP Profile. This applies to faxes sent immediately upon the establishment of a voice channel (i.e., after 200 OK). The device attempts to detect the fax (CNG tone) from the originating SIP UA within a user-defined interval (see the SBCFaxDetectionTimeout parameter) immediately after the voice call is established. Once fax is detected, the device can handle the subsequent fax negotiation by sending re-INVITE messages to both SIP entities. The device also negotiates the fax coders between the two SIP entities. The negotiated coders are according to the list of fax coders assigned to each SIP UA, using the IP Profile parameter 'Fax Coders Group'. ■ [0] **Transparent** = (Default) Device does not interfere in the fax transaction and assumes that the SIP UA fully supports fax renegotiation upon fax detection. ■ [1] **Only on Answer Side** = The SIP UA supports fax renegotiation upon fax detection only if it is the terminating (answering) fax, and does not support renegotiation if it is the originating fax. ■ [2] **No** = The SIP UA does not support fax re-negotiation upon fax detection when it is the originating or terminating fax. **Note:** ■ This feature is applicable only when both SIP entities do not fully support fax detection (receive or send) and negotiation: one SIP UA must be assigned an IP

Parameter	Description
	Profile where the parameter is set to [1] or [2], while the peer SIP UA must be assigned an IP Profile where the parameter is set to [2]. ■ This feature is supported only if at least one of the SIP entities use the G.711 coder. ■ This feature requires DSP resources. If there are insufficient resources, the fax transaction fails.
'Fax Rerouting Mode' `sbc-fax-rerouting-mode` [IpProfile_ SBCFaxReroutingMode]	Enables the rerouting of incoming SBC calls that are identified as fax calls to a new IP destination. ■ [0] **Disable** (default) ■ [1] **Rerouting without delay** For more information, see Configuring Rerouting of Calls to Fax Destinations. **Note:** Configure the parameter for the IP leg that is interfacing with the fax termination.
Media	
'Broken Connection Mode' `disconnect-on-broken-connection` [IpProfile_ DisconnectOnBrokenConnection]	Defines the device's handling of calls when RTP packets (media) are not received within a user-defined timeout. The timeout can be during call setup (configured by the [NoRTPDetectionTimeout] parameter) or mid-call when RTP flow suddenly stops (configured by the [BrokenConnectionEventTimeout] parameter). ■ [0] **Ignore** = The call is maintained despite no media and is released when signaling ends the call (i.e., SIP BYE). ■ [1] **Disconnect** = (Default) The device ends the call when the timeout expires. ■ [2] **Reroute** = The device ends the call and then searches the IP-to-IP Routing table for a matching rule. If found, the device generates a new INVITE to the corresponding destination (i.e., alternative routing). You can configure a routing rule whose matching characteristics is explicitly for calls with broken RTP connections. This is done using the 'Call Trigger' parameter, as described in Configuring SBC IP-to-IP Routing Rules.

Parameter	Description
	Note: ■ The device can only detect a broken RTP connection if silence compression is disabled for the RTP session. ■ If during a call the source IP address (from where the RTP packets are received by the device) is changed without notifying the device, the device rejects these RTP packets. To overcome this, configure the [DisconnectOnBrokenConnection] parameter to [0]. By this configuration, the device doesn't detect RTP packets arriving from the original source IP address and switches (after 300 msec) to the RTP packets arriving from the new source IP address. ■ The corresponding global parameter is [DisconnectOnBrokenConnection].
'Media IP Version Preference' `media-ip-version-preference` [IpProfile_ MediaIPVersionPreference]	Defines the preferred RTP media IP addressing version for outgoing SIP calls (according to RFC 4091 and RFC 4092). The RFCs concern Alternative Network Address Types (ANAT) semantics in the SDP to offer groups of network addresses (IPv4 and IPv6) and the IP address version preference to establish the media stream. The IP address is indicated in the "c=" field (Connection) of the SDP. ■ [0] **Only IPv4** = (Default) SDP offer includes only IPv4 media IP addresses. ■ [1] **Only IPv6** = SDP offer includes only IPv6 media IP addresses. ■ [2] **Prefer IPv4** = SDP offer includes IPv4 and IPv6 media IP addresses, but the first (preferred) media is IPv4. ■ [3] **Prefer IPv6** = SDP offer includes IPv4 and IPv6 media IP addresses, but the first (preferred) media is IPv6. To indicate ANAT support, the device uses the SIP Allow header or to enforce ANAT it uses the Require header: `Require: sdp-anat` In the outgoing SDP, each 'm=' field is associated with an ANAT group. This is done using the 'a=mid:' and 'a=group:ANAT' fields. Each 'm=' field appears under a unique 'a=mid:' number, for example:

Parameter	Description
	a=mid:1 m=audio 63288 RTP/AVP 0 8 18 101 c=IN IP6 3000::290:8fff:fe40:3e21 The 'a=group:ANAT' field shows the 'm=' fields belonging to it, using the number of the 'a=mid:' field. In addition, the ANAT group with the preferred 'm=' fields appears first. For example, the preferred group includes 'm=' fields under 'a=mid:1' and 'a=mid3': a=group:ANAT 1 3 a=group:ANAT 2 4 If you configure the parameter to a "prefer" option, the outgoing SDP offer contains two medias which are the same except for the "c=" field. The first media is the preferred address type (and this type is also on the session level "c=" field), while the second media has its "c=" field with the other address type. Both medias are grouped by ANAT. For example, if the incoming SDP contains two medias, one secured and the other non-secured, the device sends the outgoing SDP with four medias: ■ Two secured medias grouped in the first ANAT group, one with IPv4 and the other with IPv6. The first is the preferred type. ■ Two non-secured medias grouped in the second ANAT group, one with IPv4 and the other with IPv6. The first is the preferred type. **Note:** ■ The parameter is applicable only when the device offers an SDP. ■ The IP addressing version is determined according to the first SDP "m=" field. ■ The feature is applicable to any type of media (e.g., audio and video) that has an IP address. ■ The corresponding global parameter is MediaIPVersionPreference.
'RTP Redundancy Depth' rtp-redundancy-depth	Enables the device to generate RFC 2198 redundant packets. This can be used for packet loss where the missing information (audio) can be reconstructed at the

Parameter	Description
[IpProfile_ RTPRedundancyDepth]	receiver's end from the redundant data that arrives in subsequent packets. This is required, for example, in wireless networks where a high percentage (up to 50%) of packet loss can be experienced. ■ [0] **0** = (Default) Disable. ■ [1] **1** = Enable - previous voice payload packet is added to current packet. **Note:** ■ When enabled, you can configure the payload type, using the RFC2198PayloadType parameter. ■ The corresponding global parameter is RTPRedundancyDepth.
Answer Machine Detection	
'AMD Sensitivity Parameter Suite' `amd-sensitivity-parameter-suit` [IpProfile_ AMDSensitivityParameterSuit]	Defines the AMD Parameter Suite to use for the Answering Machine Detection (AMD) feature. ■ [0] **0** = (Default) Parameter Suite 0 based on North American English with standard detection sensitivity resolution (8 sensitivity levels, from 0 to 7). This AMD Parameter Suite is provided by the AMD Sensitivity file, which is shipped pre-installed on the device. ■ [1] **1** = Parameter Suite based 1 on North American English with high detection sensitivity resolution (16 sensitivity levels, from 0 to 15). This AMD Parameter Suite is provided by the AMD Sensitivity file, which is shipped pre-installed on the device. ■ [2-7] **2** to **7** = Optional Parameter Suites that you can create based on any language (16 sensitivity levels, from 0 to 15). This requires a customized AMD Sensitivity file that needs to be installed on the device. For more information, contact the sales representative of your purchased device. **Note:** ■ To configure the detection sensitivity level, use the 'AMD Sensitivity Level' parameter. ■ For more information on the AMD feature, see Answering Machine Detection (AMD).

Parameter	Description
	■ The corresponding global parameter is [AMDSensitivityParameterSuit].
'AMD Sensitivity Level' amd-sensitivity-level [IpProfile_AMDSensitivityLevel]	Defines the AMD detection sensitivity level of the selected AMD Parameter Suite (using the 'AMD Sensitivity Parameter Suite' parameter). ■ For Parameter Suite 0: The valid range is 0 to 7 (default is 0), where 0 is for best detection of an answering machine and 7 for best detection of a live call. ■ For any Parameter Suite other than 0, the valid range is 0 to 15 (default is 8), where 0 is for best detection of an answering machine and 15 for best detection of a live call. **Note:** The corresponding global parameter is [AMDSensitivityLevel].
'AMD Max Greeting Time' amd-max-greeting-time [IpProfile_AMDMaxGreetingTime]	Defines the maximum duration (in 5-msec units) that the device can take to detect a greeting message. The valid range value is 0 to 51132767. The default is 300. **Note:** The corresponding global parameter is [AMDMaxGreetingTime].
'AMD Max Post Silence Greeting Time' amd-max-post-silence-greeting-time [IpProfile_AMDMaxPostSilenceGreetingTime]	Defines the maximum duration (in 5-msec units) of silence from after the greeting time is over, configured by [AMDMaxGreetingTime], until the device's AMD decision. The valid value is 0 to 32767. The default is 400. **Note:** The corresponding global parameter is [AMDMaxPostGreetingSilenceTime].
Local Tones	
'Local Ringback Tone Index' local-ringback-tone-index [IPProfile_LocalRingbackTone]	Defines the ringback tone that you want to play from the PRT file. To associate a user-defined tone, configure the parameter with the tone's index number (1-80) as appears in the PRT file. By default (value of -1), the device plays the default ringback tone. To play user-defined tones, you need to record your tones

Parameter	Description
	and then install them on the device using a loadable Prerecorded Tones (PRT) file, which is created using AudioCodes DConvert utility. When you create the PRT file, each recorded tone file must be added to the PRT file with the tone type "acUserDefineTone<Index>". When you want to specify the ringback tone for this parameter, use the index number. For more information, see Prerecorded Tones File.
'Local Held Tone Index' `local-held-tone-index` [IPProfile_LocalHeldTone]	Defines the held tone that you want to play from the PRT file. To associate a user-defined tone, configure the parameter with the tone's index number (1-80) as appears in the PRT file. By default (value of -1), the device plays the default held tone. To play user-defined tones, you need to record your tones and then install them on the device using a loadable Prerecorded Tones (PRT) file, which is created using AudioCodes DConvert utility. When you create the PRT file, each recorded tone file must be added to the PRT file with the tone type "acUserDefineTone<Index>". When you want to specify the held tone for this parameter, use the index number. For more information, see Prerecorded Tones File.

20 SIP Definitions

This section describes configuration of various SIP-related functionality.

Configuring Registration Accounts

The Accounts table lets you configure up to 625 Accounts. An Account defines information for registering and authenticating (digest) IP Groups (e.g., IP PBX) with a "serving" IP Group (e.g., ITSP).

The device initiates registration with a "serving" IP Group on behalf of the "served" IP Group. Therefore, Accounts are typically required when the "served" IP Group is unable to register or authenticate itself for whatever reason. Registration information includes username, password, host name (AOR), and contact user name (AOR). The device includes this information in the REGISTER message sent to the serving IP Group. Up to 10 Accounts can be configured per "served" IP Group. A IP Group can register to more than one IP Group (e.g., multiple ITSPs). This is done by configuring multiple entries in the Accounts table for the same served IP Group, but with different serving IP Groups, username/password, host name, and contact user values.

You cannot configure more than one Account with the same "served" IP Group, and "serving" IP Group combination. For example, only one Account can be configured with the 'Served IP Group' parameter set to "Users-Boston" and the 'Serving IP Group' parameter set to "ITSP".

Authentication is typically required for INVITE messages sent to the "serving" IP Group. If the device receives a SIP 401 (Unauthorized) in response to a sent INVITE, the device checks for a matching "serving" and "served" entry in the Accounts table. If a matching row exists, the device authenticates the INVITE by providing the corresponding MD5 authentication username and password to the "serving" IP Group.

If the Account is not registered and the device receives a SIP dialog request (e.g., INVITE) from the Served IP Group, the device rejects the dialog and sends the Served IP Group a SIP 400 (Bad Request) response. An Account that is not registered can be due to any of the following reasons:

- You have unregistered the Served IP Group by clicking the **Register** button (discussed later in this section).
- The Serving IP Group has rejected the registration.

However, if the Account is not registered and you have enabled the Registrar Stickiness feature ('Registrar Stickiness' parameter is configured to **Enable**) or dynamic UDP port assignment feature ('UDP Port Assignment' parameter is configured to **Enable**) and the device receives a SIP dialog request (e.g., INVITE) from the Served IP Group, the device rejects the dialog and sends the Served IP Group a SIP 500 (Server Internal Error) response. In this scenario, the Account can be not registered due to any of the reasons listed previously or for the dynamic UDP port

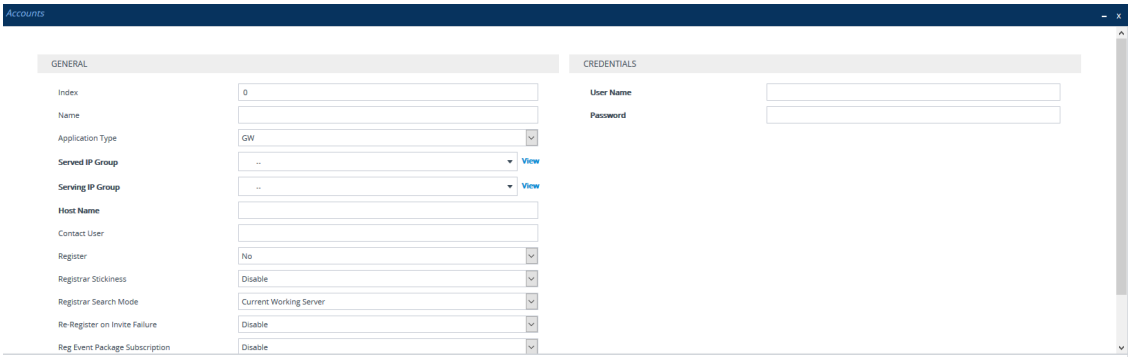
assignment feature, there is no available port for the Account (port used for interfacing with the Serving IP Group).

- SBC application: The device uses the username and password configured for the Serving IP Group in the IP Groups table for user registration and authentication, in the scenarios listed below. For this mode of operation, the 'Authentication Mode' parameter in the IP Groups table for the Serving IP Group must be configured to **SBC As Client**:
  - ✓ If there is no Account configured for the Served IP Group and Serving IP Group in the Accounts table.
  - ✓ If there is an Account configured for the Served IP Group and Serving IP Group, but without a username and password.
- See also the following optional, related parameters:
  - ✓ [UseRandomUser] - enables the device to assign a random string to the user part of the SIP Contact header of new Accounts.
  - ✓ [UnregisterOnStartup] - enables the device to unregister and then re-register Accounts upon a device reset.
  - ✓ [SyncIMSAccounts] - enables synchronization of multiple Accounts per the IMS specification.

The following procedure describes how to configure Accounts through the Web interface. You can also configure it through ini file [Account] or CLI (`configure voip > sip-definition account`).

➢ To configure an Account:

1. Open the Accounts table (**Setup** menu > **Signaling & Media** tab > **SIP Definitions** folder > **Accounts**).
2. Click **New**; the following dialog box appears:



3. Configure an account according to the parameters described in the table below.
4. Click **Apply**.

Once you have configured Accounts, you can register or un-register them, as described below:

➢ To register or un-register an Account:

1. In the table, select the required Account entry row.

2. From the **Action** drop-down list, choose one of the following commands:
   - **Register** to register the Account.
   - **Un-Register** to un-register the Account.

To view Account registration status, see Viewing Registration Status.

Table 20-1: Accounts Table Parameter Descriptions

Parameter	Description
General	
'Index'	Defines an index for the new table row. **Note:** Each row must be configured with a unique index.
'Name' `account-name` [Account_AccountName]	Defines an arbitrary name to easily identify the row. The valid value is a string of up to 20 characters. **Note:** Each row must be configured with a unique name.
'Application Type' `application-type` [Account_ApplicationType]	Defines the application type: ■ [2] **SBC** = SBC application.
'Served IP Group' `served-ip-group-name` [Account_ServedIPGroupName]	Defines the IP Group (e.g., IP-PBX) that you want to register and/or authenticate upon its behalf. **Note:** ■ The parameter is applicable only to the SBC application. ■ By default, all IP Groups are displayed. However, if you filter the Web display by SRD (using the SRD Filter box), only IP Groups associated with the filtered SRD are displayed. ■ The parameter is mandatory.
'Serving IP Group' `serving-ip-group-name` [Account_ServingIPGroupName]	Defines the IP Group (*Serving IP Group*) to where the device sends the SIP REGISTER requests (if enabled) for registration and authentication (of the Served IP Group). **Note:** ■ By default, only IP Groups associated with the SRD to which the Served IP Group is associated are displayed, as well as IP Groups of Shared SRDs.

Parameter	Description
	However, if you filter the Web display by SRD (using the SRD Filter box), only IP Groups associated with the filtered SRD are displayed, as well as IP Groups of Shared SRDs. ■ The parameter is mandatory.
'Host Name' host-name [Account_HostName]	Defines the Address of Record (AOR) host name. The host name appears in SIP REGISTER From/To headers as ContactUser@HostName. For a successful registration, the host name is also included in the URI of the INVITE From header. The valid value is a string of up to 49 characters. **Note:** If the parameter is not configured or if registration fails, the 'SIP Group Name' parameter value configured in the IP Groups table is used instead.
'Contact User' contact-user [Account_ContactUser]	Defines the AOR username. This appears in REGISTER From/To headers as ContactUser@HostName, and in INVITE/200 OK Contact headers as ContactUser@<device's IP address>. The valid value is a string of up to 60 characters. By default, no value is defined. **Note:** ■ If the parameter is not configured, the 'Contact User' parameter in the IP Groups table is used instead. ■ If registration is disabled for the Account, or registration fails, the user part in the SIP INVITE's Contact header contains the source party number. ■ If the source of the message is a registered user or matches a record in the User Information table (see Configuring SBC User Information Table through Web Interface), it has higher priority than the Account's configuration in deciding the user part in the INVITE's Contact header.
'Register' register [Account_Register]	Enables registration. ■ [0] **No** = (Default) The device only performs authentication (not registration). Authentication is typically done for INVITE messages sent to the

Parameter	Description
	"serving" IP Group. If the device receives a SIP 401 (Unauthorized) in response to a sent INVITE, the device checks for a matching "serving" and "served" entry in the table. If a matching row exists, the device authenticates the INVITE by providing the corresponding MD5 authentication username and password to the "serving" IP Group. ■ [1] **Regular** = The device performs regular registration. For more information, see Regular Registration Mode. ■ [2] **GIN** = The device performs registration for legacy PBXs, using Global Identification Number (GIN). For more information, see Single Registration for Multiple Phone Numbers using GIN. **Note:** ■ Account registration is not affected by the [IsRegisterNeeded] parameter.
'Registrar Stickiness' `registrar-stickiness` [Account_RegistrarStickiness]	Enables the Registrar Stickiness feature. ■ [0] **Disable** = (Default) Disables the Registrar Stickiness feature. After a successful initial registration of the Account with a registrar (IP address), whenever the device receives a SIP request or registration refresh, the device sends the request to whichever IP address is the currently working registrar. In other words, there is no binding to a specific IP address in the Proxy Set and at any given time, requests may be sent to a different IP address, whichever is the working one. In the case of proxy load-balancing, there is no certainty as to which IP address in the Proxy Set the request is routed. ■ [1] **Enable** = Enables the Register Stickiness feature. The device always routes SIP requests of a registered Account to the same registrar server to where the last successful REGISTER request was routed. In other words, once initial registration of the Account to one of the IP addresses in the Proxy Set (associated with the Account's Serving IP

Parameter	Description
	Group) is successful (i.e., 200 OK), binding ("stickiness") occurs to this specific address (registrar). All future SIP requests (e.g., INVITEs, SUBSCRIBEs and REGISTER refreshes) whose source and destination match the Account are sent to this registrar only. This applies until the registrar is unreachable or registration refresh fails, for whatever reason ■ [2] **Enable for Non-REGISTER Requests** = Enables the Register Stickiness feature, as described for the **Enable** option (above), except for refresh REGISTER messages. When the device initiates a refresh REGISTER message for the Account, it restarts the registration process for the Account, sending the message to one of the registrar servers according to the Proxy Set of the Account's Serving IP Group. This option can used, for example, in scenarios where proxy keep-alive is disabled (see the 'Proxy Keep-Alive' parameter in the Proxy Sets table) and restart of registration for refresh REGISTERs is always preferred. **Note:** ■ The parameter is applicable only if you have enabled Account registration ('Register' parameter configured to **Regular** or **GIN**).
'Registrar Search Mode' `registrar-search-mode` [Account_RegistrarSearchMode]	Defines the method for choosing an IP address (registrar) in the Proxy Set (associated with the Serving IP Group) to which the Account initially registers and performs registration refreshes, when the Register Stickiness feature is enabled. Once chosen, the Account is binded to the IP address for subsequent SIP requests. ■ [0] **Current Working Server** = (Default) For each initial and refresh registration request, the device routes to the currently working server in the list of IP addresses (configured or DNS-resolved IP addresses) in the Proxy Set. In the case of proxy load-balancing, the chosen IP address is according to the load-balancing mechanism.

Parameter	Description
	■ [1] **According to IMS Specifications** = For the initial registration request, the device performs DNS resolution if the address of the Proxy Set is configured as an FQDN. It then attempts to register to one of the listed DNS-resolved addresses (or configured IP addresses), starting with the first listed address and then going down the list sequentially. If an address results in an unsuccessful registration, the device immediately tries the next address (without waiting any retry timeout). The device goes through the list of addresses until an address results in a successful registration. If registration is unsuccessful for all addresses, the device waits a configured retry time and then goes through the list again. Once initial registration is successful, periodic registration refreshes are performed as usual. In addition to the periodic refreshes, immediate register refreshes are done upon the following triggers according to the IMS specification: ✓ The device receives a SIP 408, 480, or 403 response from the Serving IP Group in response to an INVITE. ✓ The transaction timeout expires for an INVITE sent to the Serving IP Group. ✓ The device receives an INVITE from the Serving IP Group from an IP address other than the address to which it is currently registered. In this case, it also rejects the INVITE with a SIP 480 response. If the device's physical Ethernet link to the proxy goes down, the device re-registers this Account with the proxy when the link comes up again. Re-registration occurs even if proxy keep-alive is disabled. **Note:** This option is applicable only if you have configured the following: ✓ 'Register' parameter to **Regular** or **GIN**. ✓ 'Registrar Stickiness' parameter to **Enable**.

Parameter	Description
	You can also configure synchronization between multiple Accounts per IMS specifications. For more information. see Synchronizing Multiple SIP Accounts per IMS Specification on page 584.
'Re-REGISTER on INVITE Failure' re-register-on-invite-failure [Account_ReRegisterOnInviteFailure]	Enables the device to re-register an Account upon the receipt of specific SIP response codes (e.g., 403, 408, and 480) for a failed INVITE message sent to the Serving IP Group. ■ [0] **Disable** = (Default) If the device receives a SIP response for a failed INVITE message, the device does not re-register the Account. ■ [1] **Enable** = If the device receives a SIP response for a failed INVITE message and the response code is configured in the global parameter, AccountInviteFailureTriggerCodes, the device re-registers the Account according to the settings of the Proxy Set associated with the Account's Serving IP Group. Note that if the Proxy Set's 'Proxy Hot Swap' parameter is configured to **Enable** and the 'Proxy Keep-Alive' parameter is enabled, the registrar at which the INVITE failed is tried last in the list of servers in the Proxy Set.
'Reg Event Package Subscription' reg-event-package-subscription [Account_RegEventPackageSubscription]	Enables the device to subscribe to the registration event package service (as defined in RFC 3680) with the registrar server (Serving IP Group) to which the Account is successfully registered and binded, when the Registrar Stickiness feature is enabled. The service allows the device to receive notifications of the Accounts registration state change with the registrar. The device subscribes to the service by sending a SUBSCRIBE message containing the Event header with the value "reg" (Event: reg). Whenever a change occurs in the registration binding state, the registrar notifies the device by sending a SIP NOTIFY message. ■ [0] **Disable** (default) ■ [1] **Enable** **Note:** The parameter is applicable only if you have enabled the Registrar Stickiness feature (in this table):

Parameter	Description
	■ 'Register' parameter to **Regular** or **GIN**. ■ 'Registrar Stickiness' parameter to **Enable**.
'Register by Served IP Group Status' `reg-by-served-ipg-status` [Account_RegByServedIPG]	Defines the device's handling of Account registration based on the connectivity status of the Served IP Group. ■ [0] **Register Always** = (Default) Account registration by the device does not depend on the connectivity status of the Served IP Group. The device sends registration requests to the Serving IP Group even if the Served IP Group is offline. ■ [1] **Register Only if Online** = The device performs Account registration depending on the connectivity status of the Served IP Group. It sends a registration request to the Serving IP Group only if the Served IP Group is online. If the Served IP Group was registered, but then goes offline, the device unregisters it. If it becomes online again, the device re-registers it. This option is applicable only to Accounts where registration is initiated by the device (i.e., the 'Register' parameter is configured to any value other than **No**). The Served IP Group's connectivity status is determined by the keep-alive mechanism of its associated Proxy Set (i.e., the 'Proxy Keep-Alive' parameter is configured to **Using OPTIONS** or **Using OPTIONS on Active Server**.
'UDP Port Assignment' `udp-port-assignment` [Account_UDPPortAssignment]	Enables the device to dynamically allocate local SIP UDP ports to Accounts using the same Serving IP Group, where each Account is assigned a unique port on the device's leg interfacing with the Accounts' Serving IP Group. ■ [0] **Disable** = (Default) The device uses the same specific UDP port for all registrations done for this Account (traffic between the device and the Serving IP Group). This port is the one configured for the SIP Interface ('UDP Port' parameter - SIPInterface_UDPPort) that is associated with the Proxy Set of the Account's Serving IP Group.

Parameter	Description
	■ [1] **Enable** = The device assigns a unique local port for each Account for which the device initiates registration. The port is taken from a configured UDP port range. The port range is configured for the SIP Interface ('Additional UDP Ports' parameter - SIPInterface_AdditionalUDPPorts) associated with the Proxy Set of the Account's Serving IP Group. Traffic between the Serving IP Group and device is sent from and received on the assigned unique local port. If enabled for other Accounts that are configured with the same Serving IP Group, each Account is allocated a unique UDP port from the port range. For example, if you have configured two Accounts, "PBX-1" and "PBX-2", the device could assign port 6000 to "PBX-1" and 6100 to "PBX-2". **Note:** ■ If you enable the parameter, you must also enable the device to initiate registration for the Account (i.e., configure the 'Register' parameter to any value other than **No**). ■ If the device fails to allocate a port (e.g., insufficient ports), the device does not send the SIP REGISTER request, but tries again within a period configured by the RegistrationRetryTime and MaxRegistrationBackoffTime parameters. ■ If the device receives a SIP request from the Serving IP Group for the Account, on a port that was not assigned to the Account, it rejects the request (with a SIP 404 Not Found response). ■ If the device receives a SIP request from the Served IP Group and the Account has not been allocated a valid port, the device rejects the request (with a SIP 500 Server Internal Error response). ■ For more information on configuring the SIP Interface's port range, see Configuring SIP Interfaces on page 434.
Credentials	

Parameter	Description
'User Name' user-name [Account_Username]	Defines the digest MD5 Authentication username. The valid value is a string of up to 60 characters. By default, no value is defined.
'Password' password [Account_Password]	Defines the digest MD5 Authentication password. The valid value is a string of up to 50 characters. **Note:** ■ The password cannot be configured with wide characters. ■ If the password contains a question mark (?) and you're configuring the parameter through CLI, you must enclose the entire password in double quotation marks (e.g., "43LSyk+?").

Regular Registration Mode

When you configure the registration mode ('Register') in the Accounts table to **Regular**, the device sends REGISTER requests to the Serving IP Group. The host name (in the SIP From/To headers) and contact user (user in From/To and Contact headers) are taken from the configured Accounts table upon successful registration. See the example below:

```
REGISTER sip:xyz SIP/2.0
Via: SIP/2.0/UDP 10.33.37.78;branch=z9hG4bKac1397582418
From: <sip:ContactUser@HostName>;tag=1c1397576231
To: <sip: ContactUser@HostName >
Call-ID: 1397568957261200022256@10.33.37.78
CSeq: 1 REGISTER
Contact: <sip:ContactUser@10.33.37.78>;expires=3600
Expires: 3600
User-Agent: Sip-Gateway/7.20A.258.980
Content-Length: 0
```

Single Registration for Multiple Phone Numbers using GIN

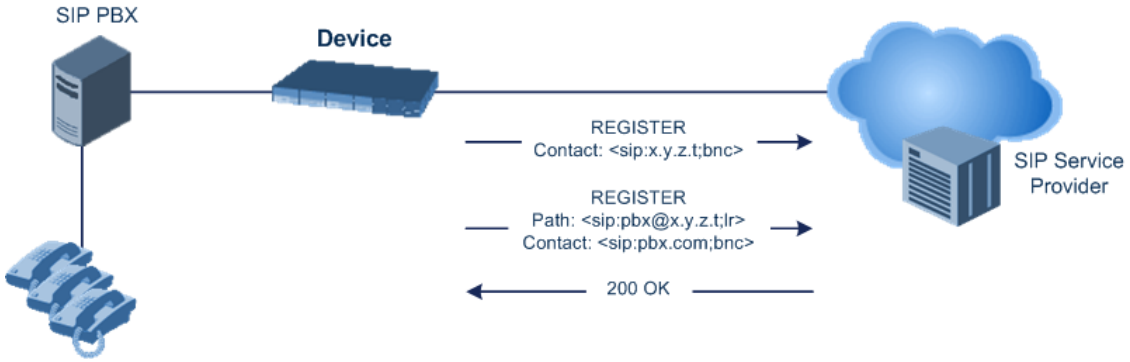
When you configure the registration mode in the Accounts table to **GIN**, the Global Identifiable Number (GIN) registration method is used, according to RFC 6140. The device performs GIN-based registration of users to a SIP registrar on behalf of a SIP PBX. In effect, the PBX registers with the service provider, just as a directly hosted SIP endpoint would register. However, because a PBX has multiple user agents, it needs to register a contact address on behalf of each of these. Rather than performing a separate registration procedure for each user agents, GIN registration mode does multiple registrations using a single REGISTER transaction.

According to this mechanism, the PBX delivers to the service provider in the Contact header field of a REGISTER request a template from which the service provider can construct contact URIs for each of the AORs assigned to the PBX and thus, can register these contact URIs within its location service. These registered contact URIs can then be used to deliver to the PBX inbound requests targeted at the AORs concerned. The mechanism can be used with AORs comprising SIP URIs based on global E.164 numbers and the service provider's domain name or sub-domain name.

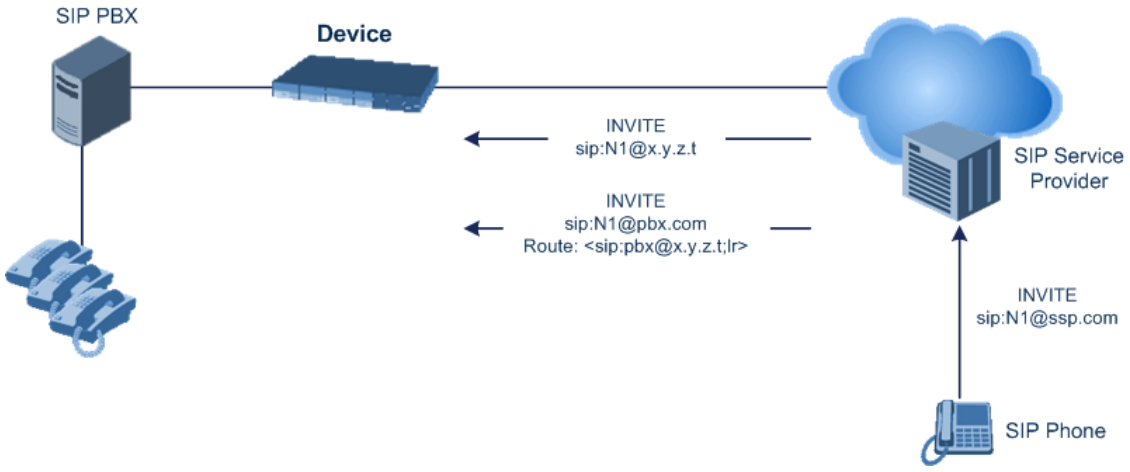
The SIP REGISTER request sent by the device for GIN registration with a SIP server provider contains the Require and Proxy-Require headers. These headers contain the token 'gin'. The Supported header contains the token 'path' and the URI in the Contact header contains the parameter 'bnc' without a user part:

```
Contact: <sip:198.51.100.3;bnc>;
```

The figure below illustrates the GIN registration process:



The figure below illustrates an incoming call using GIN:



Synchronizing Multiple SIP Accounts per IMS Specification

You can enable synchronization between multiple Accounts according to the IMS specification.

The first Account (lowest index number) that is configured for the IMS specification (see procedure below) is considered the "primary" Account. All other Accounts that are configured

for the IMS specification are considered as "secondary" Accounts. All the Accounts must have the same Serving IP Group. Up to 99 "secondary" Accounts are supported.

Synchronization between Accounts mainly concerns the registrar server that is used by the Accounts. All Accounts send all associated requests (SIP REGISTERS for the Accounts themselves, and calls matched to the Accounts) to the same single server. The "primary" Account determines the server to use.

Only the "primary" Account does the full "By IMS Specification" registration process. It triggers DNS-resolution of the Proxy Set of the Serving IP Group if the Proxy Set is configured with an FQDN host. It also attempts to register to each of the resolved registrar servers (IP addresses) of the Proxy Set, until it succeeds.

Once the "primary" Account succeeds in registering to a server, the "secondary" Accounts that are enabled for registration ('Register' parameter configured to **Regular**), register to this same server (and do not attempt to register to any other server). If the "primary" Account is not registered, the "secondary" Accounts can't send REGISTER requests (unless they were already registered prior to the "primary" Account's failure; then they may continue working with the last server as long as it accepts their refresh REGISTERs).

If the "primary" Account registers to a new server (e.g., it was registered to the first address in the Proxy Set and because registration refresh subsequently failed, it registered to the second address in the Proxy Set), the "secondary" Accounts then automatically register to this new server.

Registration failures of "secondary" Accounts trigger the "primary" Account to do an immediate refresh registration. (Only if the refresh REGISTERs of the "primary" Account fail, does the "primary" Account start the full registration process.)

Triggers for immediate refresh registrations, as mandated by the IMS specification (e.g., triggered by failure of outgoing INVITE) are handled normally by "secondary" Accounts that are enabled for registration. "Secondary" Accounts that are disabled for registration ('Register' parameter configured to **No**), forward the trigger to the "primary" Account, and the "primary" Account sends a refresh REGISTER instead of them.

➢ **To enable synchronization of multiple Accounts per IMS:**

1. Enable the feature by configuring the [SyncIMSAccounts] global parameter to 1.
2. In the Accounts table, configure all the Accounts for synchronization per IMS with the following settings:
   - 'Registrar Search Mode': **By IMS Specification**
   - 'Serving IP Group': <same IP Group>

Registrar Stickiness

You can enable the Registrar Stickiness feature per Account. Registrar Stickiness binds an Account to one of the IP addresses (configured or DNS-resolved) in the Proxy Set associated with the Serving IP Group. Once an Account registers successfully to one of the IP addresses

(i.e., SIP registrar server) in the Proxy Set, the device routes all subsequent SIP requests (INVITEs, SUBSCRIBEs and REGISTER refreshes) of the Account to this registrar. This applies until the registrar is unreachable or registration refresh fails, for whatever reason.

To configure the Registrar Stickiness feature, use the following parameters in the Accounts table:

- Registrar Stickiness: Enables the feature.
- Registrar Search Mode: Defines the method for choosing an IP address (registrar) in the Proxy Set to which the Account initially registers and performs registration refreshes. Once chosen, the Account is binded to this registrar.
- Reg Event Package Subscription: Enables the device to subscribe to the registration event package service (as defined in RFC 3680) with the registrar to which the Account is registered and binded. The service allows the device to receive notifications of the Accounts registration state change with the registrar.

Configuring Proxy and Registration Parameters

The Proxy & Registration page allows you to configure the Proxy server and registration parameters. For a description of the parameters appearing on this page, see Configuration Parameters Reference. To configure Proxy servers (Proxy Sets), see Configuring Proxy Sets.

To view the registration status of endpoints with a SIP Registrar/Proxy server, see Viewing Registration Status.

➢ To configure the Proxy and registration parameters:

1. Open the Proxy & Registration page (**Setup** menu > **Signaling & Media** tab > **SIP Definitions** folder > **Proxy & Registration**).
2. Configure the parameters as required.
3. Click **Apply**.

➢ To register or un-register the device to a Proxy/Registrar:

- Click the **Register** button to register.
- Click **Un-Register** button to un-register.

Instead of registering the entire device, you can register specific entities as listed below by using the **Register** button located on the page in which these entities are configured:

- Trunk Groups - Trunk Group table (see Configuring Trunk Groups)
- Accounts - Accounts table (see Configuring Registration Accounts)

SIP Message Authentication Example

The device supports basic and digest (MD5) authentication types, according to SIP RFC 3261. A proxy server might require authentication before forwarding an INVITE message. A Registrar/Proxy server may also require authentication for client registration. A proxy replies to an unauthenticated INVITE with a 407 Proxy Authorization Required response, containing a Proxy-Authenticate header with the form of the challenge. After sending an ACK for the 407, the user agent can then re-send the INVITE with a Proxy-Authorization header containing the credentials.

User agents, Redirect or Registrar servers typically use the SIP 401 Unauthorized response to challenge authentication containing a WWW-Authenticate header, and expect the re-INVITE to contain an Authorization header.

The following example shows the Digest Authentication procedure, including computation of user agent credentials:

1. The REGISTER request is sent to a Registrar/Proxy server for registration:

```
REGISTER sip:10.2.2.222 SIP/2.0
Via: SIP/2.0/UDP 10.1.1.200
From: <sip: 122@10.1.1.200>;tag=1c17940
To: <sip: 122@10.1.1.200>
Call-ID: 634293194@10.1.1.200

CSeq: 1 REGISTER
Contact: sip:122@10.1.1.200:
Expires:3600
```

2. Upon receipt of this request, the Registrar/Proxy returns a 401 Unauthorized response:

```
SIP/2.0 401 Unauthorized
Via: SIP/2.0/UDP 10.2.1.200
From: <sip:122@10.2.2.222 >;tag=1c17940
To: <sip:122@10.2.2.222 >
Call-ID: 634293194@10.1.1.200
Cseq: 1 REGISTER
Date: Mon, 30 Jul 2012 15:33:54 GMT
Server: Columbia-SIP-Server/1.17
Content-Length: 0
```

```
WWW-Authenticate: Digest realm="AudioCodes.com",
nonce="11432d6bce58ddf02e3b5e1c77c010d2",
stale=FALSE,
algorithm=MD5
```

3. According to the sub-header present in the WWW-Authenticate header, the correct REGISTER request is created.
4. Since the algorithm is MD5:
   - The username is equal to the endpoint phone number "122".
   - The realm return by the proxy is "AudioCodes.com".
   - The password from the *ini* file is "AudioCodes".
   - The equation to be evaluated is "122:AudioCodes.com:AudioCodes". According to the RFC, this part is called A1.
   - The MD5 algorithm is run on this equation and stored for future usage.
   - The result is "a8f17d4b41ab8dab6c95d3c14e34a9e1".
5. The par called A2 needs to be evaluated:
   - The method type is "REGISTER".
   - Using SIP protocol "sip".
   - Proxy IP from *ini* file is "10.2.2.222".
   - The equation to be evaluated is "REGISTER:sip:10.2.2.222".
   - The MD5 algorithm is run on this equation and stored for future usage.
   - The result is "a9a031cfddcb10d91c8e7b4926086f7e".
6. Final stage:
   - A1 result: The nonce from the proxy response is "11432d6bce58ddf02e3b5e1c77c010d2".
   - A2 result: The equation to be evaluated is "A1:11432d6bce58ddf02e3b5e1c77c010d2:A2".
   - The MD5 algorithm is run on this equation. The outcome of the calculation is the response needed by the device to register with the Proxy.
   - The response is "b9c45d0234a5abf5ddf5c704029b38cf".

   At this time, a new REGISTER request is issued with the following response:

```
REGISTER sip:10.2.2.222 SIP/2.0
Via: SIP/2.0/UDP 10.1.1.200
From: <sip: 122@10.1.1.200>;tag=1c23940
To: <sip: 122@10.1.1.200>
Call-ID: 654982194@10.1.1.200

CSeq: 1 REGISTER
Contact: sip:122@10.1.1.200:
Expires:3600
```

```
Authorization: Digest, username: 122,
realm="AudioCodes.com”,
nonce="11432d6bce58ddf02e3b5e1c77c010d2",
uri=”10.2.2.222”,
response=“b9c45d0234a5abf5ddf5c704029b38cf”
```

7. Upon receiving this request and if accepted by the Proxy, the Proxy returns a 200 OK response, completing the registration transaction:

```
SIP/2.0 200 OK
Via: SIP/2.0/UDP 10.1.1.200
From: <sip: 122@10.1.1.200>;tag=1c23940
To: <sip: 122@10.1.1.200>
Call-ID: 654982194@10.1.1.200
Cseq: 1 REGISTER
Date: Thu, 26 Jul 2012 09:34:42 GMT
Server: Columbia-SIP-Server/1.17
Content-Length: 0
```

```
Contact: <sip:122@10.1.1.200>; expires="Thu, 26 Jul 2012 10:34:42 GMT";
action=proxy; q=1.00
```

```
Contact: <122@10.1.1.200:>; expires="Tue, 19 Jan 2038 03:14:07 GMT";
action=proxy; q=0.00
```

```
Expires: Thu, 26 Jul 2012 10:34:42 GMT
```

Configuring User Information

This section describes User Information configuration.

Enabling the User Information Table

Before you can use the User Information table, you need to enable the User Information functionality.

➢ **To enable User Information functionality:**

1. Make sure that your device's License Key includes the far-end user license ("Far End Users"), which specifies the maximum number of supported users. To view the License Key, see Viewing the License Key.

2. Open the Proxy & Registration page (**Setup** menu > **Signaling & Media** tab > **SIP Definitions** folder > **Proxy & Registration**).
3. From the 'User-Information Usage' drop-down list [EnableUserInfoUsage], select **Enable**:

4. Reset the device with a save-to-flash for your settings to take effect; the User Information table now becomes available in the Web interface.

Configuring SBC User Information

The User Information table lets you configure up to or 20,000 SBC users. You can use the table for the following:

- Registering each user to an external registrar server.
- Authenticating (for any SIP request and as a client) each user if challenged by an external server.
- Authenticating as a server incoming user requests (for SBC security).

If the device registers on behalf of users and the users do not perform registration, any SIP request destined to the user is routed to the Proxy Set associated with the user's IP Group.

The User Information table can be configured using any of the following methods:

- Web interface (see Configuring SBC User Info Table through Web Interface)
- CLI (see Configuring SBC User Info Table through CLI)
- Loadable User Information file (see Configuring SBC User Info Table in Loadable Text File)

- For the SBC User Information feature, the device's License Key must include the license "Far End Users (FEU)", which specifies the maximum number of supported far-end users. If no far-end users are licensed, then this feature cannot be used.
- If you configure the device to authenticate as a server the incoming SIP requests from users of a specific User-type IP Group, the device authenticates the users, using the username and password configured in the IP Group's 'Username' and 'Password' parameters. However, if the user appears in the User Information table and configured with a username and/or password, then the device authenticates the user with the credentials in the table. To enable the device to authenticate as a server, configure the IP Group's 'Authentication Mode' parameter to **SBC as Server**.

Configuring SBC User Information Table through Web Interface

You can configure the User Information table for SBC users through the Web interface. The table allows you to do the following:

- Manually add users (described below).

- Import users from a file: From the **Action** drop-down list, choose **Import**.

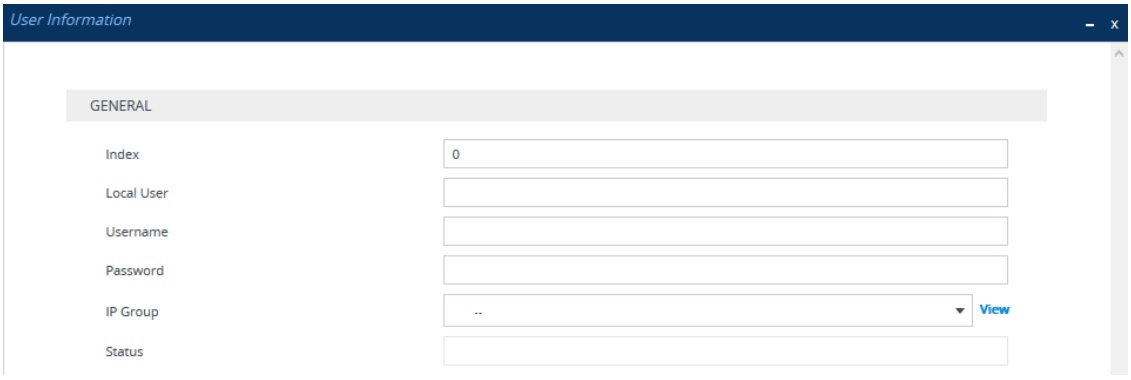
> - When you import a file, all previously configured entries in the table are deleted and replaced with the users from the imported file.
> - For configuring users in a file for import, see Configuring SBC User Information Table from a Loadable File on page 594.

- Export the configured users to a file (.csv file format): From the **Action** drop-down list, choose **Export** and save the file to a folder on your computer.
- Register and un-register users:
  - To register a user: Select the user, and then from the **Action** drop-down list, choose **Register**.
  - To un-register a user: Select the user, and then from the **Action** drop-down list, choose **Un-Register**.

> To configure the User Information table, make sure that you have enabled the feature (see Enabling the User Info Table).

➢ To configure User Information table through the Web interface:

1. Open the User Information table (**Setup** menu > **Signaling & Media** tab > **SBC** folder > **User Information**).
2. Click **New**; the following dialog box appears:



3. Configure a user according to the table below.
4. Click **Apply**.

Table 20-2: User Information Table Parameter Descriptions

Parameter	Description
'Index' [SBCUserInfoTable_	Defines an index number for the new table row. **Note:** Each row must be configured with a unique index.

Parameter	Description
Index]	
'Local User' [SBCUserInfoTable_LocalUser]	Defines the user and is used as the Request-URI user part for the AOR in the database. The valid value is a string of up to 60 characters. By default, no value is defined. **Note:** The parameter is mandatory.
'Username' [SBCUserInfoTable_Username]	Defines the username for registering the user when authentication is necessary. The valid value is a string of up to 60 characters. By default, no value is defined.
'Password' [SBCUserInfoTable_Password]	Defines the password for registering the user when authentication is necessary. The valid value is a string of up to 20 characters. **Note:** The password cannot be configured with wide characters.
'IP Group' [SBCUserInfoTable_IPGroupName]	Assigns an IP Group to the user. The IP Group is used as the Request-URI source host part for the AOR in the database. To configure IP Groups, see Configuring IP Groups. **Note:** ■ The parameter is mandatory. ■ You must assign the user with a User-type IP Group.
'Status' [SBCUserInfoTable_Status]	(Read-only field) Displays the status of the user: ■ "Registered": Valid configuration and the user is registered. ■ "Not Registered": Valid configuration but the user has not been registered. ■ "N/A": Invalid configuration as the user has not been assigned an IP Group. ■ "NA": Invalid configuration as the user has been assigned a Server-type IP Group instead of a User-type IP Group.

Configuring SBC User Information Table through CLI

The SBC User Information table can be configured in the CLI using the following commands:

■ To add and/or modify a user (example):

```
# configure voip
(config-voip)# sip-definition proxy-and-registration
(sip-def-proxy-and-reg)# user-info sbc-user-info <index, e.g., 1>
(sbc-user-info-1)# username JohnDee
(sbc-user-info-1)# <activate | exit>
```

- To delete a specific user, use the no command:

```
(sip-def-proxy-and-reg)# no user-info sbc-user-info <index, e.g., 1>
```

- To import users from a file:

```
# configure voip
(config-voip)# sip-definition proxy-and-registration
(sip-def-proxy-and-reg)# user-info sbc-user-info import-csv-from <URL>
```

- To export users to a .csv file:

```
# configure voip
(config-voip)# sip-definition proxy-and-registration
(sip-def-proxy-and-reg)# user-info sbc-user-info export-csv-to <URL>
```

- To view all table entries:

```
(sip-def-proxy-and-reg)# user-info sbc-user-info display
---- sbc-user-info-0 ----
 local-user (JohnDee)
 username (userJohn)
 password (s3fn+fn=)
 ip-group-id (1)
 status (not-resgistered)
```

```
---- sbc-user-info-1 ----
 local-user (SuePark)
 username (userSue)
 password (t6sn+un=)
 ip-group-id (1)
 status (not-resgistered)
```

- To view a specific entry (example):

```
(sip-def-proxy-and-reg)# user-info sbc-user-info <index, e.g., 0>
(sbc-user-info-0)# display
```

```
local-user (JohnDee)
username (userJohn)
password (s3fn+fn=)
ip-group-id (1)
status (not-resgistered)
```

- To search a user by local-user:

```
(sip-def-proxy-and-reg)# user-info find <local-user, e.g., JohnDoe>
JohnDee: Found at index 0 in SBC user info table, not registered
```

> To configure the User Information table, make sure that you have enabled the feature as described in Enabling the User Info Table.

Configuring SBC User Information Table from a Loadable File

You can configure users in a file and then load (import) it to the SBC User Information table. The users must be configured in comma-separated value (CSV) file format. You can create the file using any standard text-based editor such as Notepad, or alternatively a CSV-based program such as Microsoft Excel. The file can have any filename extension (e.g., .csv or .txt).

> - When you import a file, all previously configured entries in the table are deleted and replaced with the users from the imported file.
> - If a user is configured in the file with an IP Group that does not exist, the user is not assigned an IP Group when you import the file.

When adding users to the file, use the following syntax:

- For text-based editors:

```
LocalUser,UserName,Password,IPGroupName
```

For example:

```
LocalUser,UserName,Password,IPGroupName
John,johnd,2798,ITSP
Sue,suep,1234,IP-PBX
```

- For CSV-based programs:

```
LocalUser,UserName,Password,IPGroupName
```

For example:

	A	B	C	D
1	LocalUser	Username	Password	IPGroupName
2	John	johnd	2798	ITSP
3	Sue	suep	1234	IP-PBX
4				

You can load the User Information file using any of the following methods:

- Web interface - User Information table (see Configuring SBC User Information Table through Web Interface on page 590)
- CLI - **sbc user-info-table import-csv-from** (see Configuring SBC User Information Table through CLI on page 592)
- Automatic Update mechanism - [SBCUserInfoFileUrl] parameter (see Automatic Update Mechanism)

For **backward compatibility** only: When configuring a User Information file to load through the Auxiliary Files page, use the following syntax:
[SBC]
FORMAT LocalUser,UserName,Password,IPGroupID
For example:
[SBC]
FORMAT LocalUser,UserName,Password,IPGroupID
John,johnd,2798,2
Sue,suep,1234,1

Configuring Call Setup Rules

The Call Setup Rules table lets you configure up to 64 Call Setup rules. Call Setup rules define various sequences that are run upon the receipt of an incoming call (dialog) at call setup, before the device routes the call to its destination. Call Setup rules provide you with full flexibility in implementing simple or complex script-like rules that can be used for Lightweight Directory Access Protocol (LDAP) based routing as well as other advanced routing logic requirements such as manipulation. These Call Setup rules are assigned to routing rules.

Below is a summary of functions for which you can employ Call Setup rules:

- LDAP queries: LDAP is used by the device to query Microsoft's Active Directory (AD) server for specific user details for routing, for example, office extension number, mobile number, private number, OCS (Skype for Business) address, and display name. Call Setup rules provides full flexibility in AD-lookup configuration to suit just about any customer deployment requirement:
  - Routing based on query results.
  - Queries based on any AD attribute.

- Queries based on any attribute value (alphanumeric), including the use of the asterisk (*) wildcard as well as the source number, destination number, redirect number, and SBC SIP messages. For example, the following Call Setup rule queries the attribute "proxyAddresses" for the record value "WOW:" followed by source number: "proxyAddresses=WOW:12345*"
- Conditional LDAP queries, for example, where the query is based on two attributes (& (telephoneNumber=4064)(company=ABC)).
- Conditions for checking LDAP query results.
- Manipulation of call parameters such as source number, destination number, and redirect number and SBC SIP messages, while using LDAP query results.
- Multiple LDAP queries.

- Dial Plan queries: For SBC calls, you can use Call Setup rules to query the Dial Plan table (see Configuring Dial Plans) for a specified key in a specified Dial Plan to obtain the corresponding Dial Plan tag. Call Setup rules can also change (modify) the name of the obtained tag. The device can then route the call using an IP-to-IP Routing rule (in the IP-to-IP Routing table) that has a matching tag (source or destination). You can also associate a Call Setup rule with an IP Group (in the IP Group table). Once the device classifies the incoming call to a source IP Group, it processes the associated Call Setup rule and then uses the resultant tag to locate a matching IP-to-IP Routing rule. You can also use Call Setup rules for complex routing schemes by using multiple Dial Plan tags. This is typically required when the source or destination of the call needs to be categorized with more than one characteristics. For example, tags can be used to categorize calls by department (source user) within a company, where only certain departments are allowed to place international calls.
- ENUM queries: For SBC calls, you can use Call Setup rules to query an ENUM server and to handle responses from the ENUM server. ENUM translates ordinary telephone numbers (E.164 telephone numbers) into Internet addresses (SIP URIs), using the ENUM's DNS NAPTR records. For example, if the device receives an INVITE message whose destination number is in E.164 format, you can configure a Call Setup rule to query the ENUM server for the corresponding URI address, which is then used in the INVITE's Request-URI.
- HTTP requests (queries): You can use Call Setup rules to query or notify an HTTP/S server, which is configured in the Remote Web Services table (Configuring Remote Web Services on page 316). If a response is expected from the server, the query is sent as an HTTP GET or HTTP POST request (configurable). If no response is required from the server (i.e., to notify the server of a specific condition), then an HTTP POST for notifications is sent (configurable).
- Manipulation (similar to the Message Manipulations table) of call parameters (such as source number, destination number, and redirect number) and SBC SIP messages.
- Conditions for routing, for example, if the source number equals a specific value, then use the call routing rule.

You configure multiple Call Setup rules and group them using a *Set ID*. This lets you apply multiple Call Setup rules on the same call setup dialog. To use your Call Setup rule(s), you need to assign the Set ID to one of the following, using the 'Call Setup Rules Set ID' field:

- SBC IP-to-IP routing rules (see Configuring SBC IP-to-IP Routing Rules)
- SIP Interface rules (see Configuring SIP Interfaces on page 434)
- IP Groups (see Configuring IP Groups)

If assigned to an IP Group, the device processes the Call Setup rule for the classified source IP Group immediately before the routing process. If assigned to a routing rule only, the device first locates a matching routing rule for the incoming call, processes the assigned Call Setup Rules Set ID, and then routes the call according to the destination configured for the routing rule. The device uses the routing rule to route the call depending on the result of the Call Setup Rules Set ID:

- **Rule's condition is met:** The device performs the rule's action, and then runs the next rule in the Set ID until the last rule or until a rule whose 'Action Type' parameter is configured to **Exit**. If this "exit" rule is also configured with a **True** value for the 'Action Value' parameter, the device uses the current routing rule. If this "exit" rule is configured with a **False** value for the 'Action Value' parameter, the device moves to the next routing rule. If the 'Action Type' parameter is not configured to **Exit** and the device has run all the rules in the Set ID, the default of the 'Action Value' parameter of the Set ID is **True** (i.e., use current routing rule).
- **Rule's condition is not met:** The device runs the next rule in the Set ID. When the device reaches the end of the Set ID and no "exit" was performed, the Set ID ends with a "true" result.

You can also configure a Call Setup rule that determines whether the device must discontinue with the Call Setup Rules Set ID and route the call accordingly. This is done using the **Exit** optional value of the ‘Action Type’ parameter. When used, the ‘Action Value’ parameter can be configured to one of the following:

- **True**: Indicates that if the condition is met, the device routes the call according to the selected routing rule. Note that if the condition is not met, the device also uses the selected routing rule, unless the next Call Setup rule in the Set ID has an **Exit** option configured to **False** for an empty condition.
- **False**: Indicates that if the condition is met, the device attempts to route the call to the next matching routing rule (if configured). If the condition is not met, the device routes the call according to the selected routing rule.

As the default result of a Call Setup rule is always “true” (**True**), please adhere to the following guidelines when configuring the ‘Action Type’ field to **Exit**: If, for example, you want to exit the Call Setup Rule Set ID with "true" when LDAP query result is found and "false" when LDAP query result is not found:

- Incorrect: This rule always exits with result as "true":

  'Condition': ldap.found exists

'Action Type': **Exit**

'Action Value': **True**

- Correct:
  - Single rule:

    'Condition': ldap.found !exists

    'Action Type': **Exit**

    'Action Value': **False**

  - Set of rules:

    'Condition': ldap.found exists

    'Action Type': **Exit**

    'Action Value': **True**

    'Condition': <leave empty>

    'Action Type': **Exit**

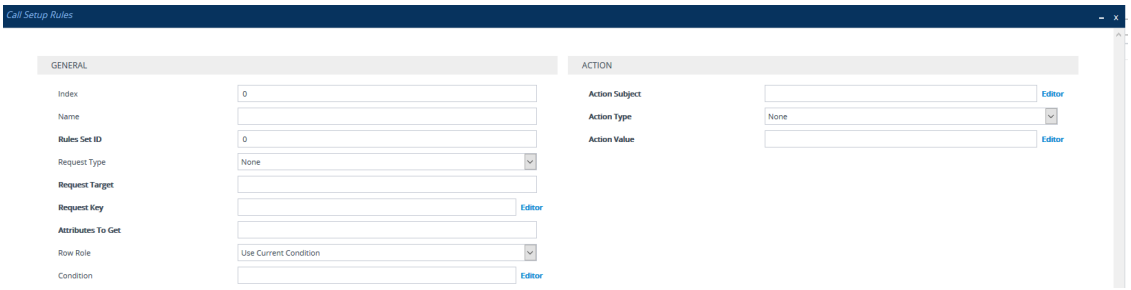
    'Action Value': **False**

> If the source or destination numbers are manipulated by the Call Setup rules, they revert to their original values if the device moves to the next routing rule.

The following procedure describes how to configure Call Setup Rules through the Web interface. You can also configure it through ini file [CallSetupRules] or CLI (`configure voip > message call-setup-rules`).

➢ To configure a Call Setup rule:

1. Open the Call Setup Rules table (**Setup** menu > **Signaling & Media** tab > **SIP Definitions** folder > **Call Setup Rules**).
2. Click **New**; the following dialog box appears:



3. Configure a Call Setup rule according to the parameters described in the table below.
4. Click **Apply**, and then save your settings to flash memory.

Table 20-3: Call Setup Rules Parameter Descriptions

Parameter	Description
General	
'Index' [CallSetupRules_Index]	Defines an index number for the new table record. **Note:** Each rule must be configured with a unique index.
'Name' `rules-set-name` [CallSetupRules_RulesSetName]	Defines an arbitrary name to easily identify the row. The valid value is a string of up to 20 characters. **Note:** Each row must be configured with a unique name.
'Rules Set ID' `rules-set-id` [CallSetupRules_RulesSetID]	Defines a Set ID for the rule. You can define the same Set ID for multiple rules to create a group of rules. You can configure up to 32 Set IDs, where each Set ID can include up to 10 rules. The Set ID is used to assign the Call Setup rules to a routing rule in the routing table. The valid value is 0 to 31. The default is 0.
'Request Type' `request-type` [CallSetupRules_QueryType]	Defines the type of query. ■ [0] **None** (default) ■ [1] **LDAP** = The Call Setup rule performs an LDAP query with an LDAP server. To specify an LDAP server, use the 'Request Target' parameter (see below). ■ [2] **Dial Plan** = The Call Setup rule performs a query with the Dial Plan. To specify a Dial Plan, use the 'Request Target' parameter (see below). ■ [3] **ENUM** = The Call Setup rule performs a query with an ENUM (E.164 Number to URI Mapping) server for retrieving a SIP URI address for an E.164 telephone number. The ENUM server's address is the address configured for the 'Primary DNS' (and optionally, 'Secondary DNS') parameters of the IP Interface (in the IP Interfaces table) that is specified in the 'Request Target' parameter (see below). For a configuration example, see Call Setup Rule Examples on page 605. ■ [4] **HTTP GET** = The Call Setup rule performs an HTTP GET request (query) with an HTTP/S server. To specify an HTTP server, use the 'Request Target' parameter (see below). ■ [5] **HTTP POST Query** = The Call Setup rule sends an HTTP POST request (query) to an HTTP/S server and expects a

Parameter	Description
	response from the server. To specify an HTTP server, use the 'Request Target' parameter (see below). ■ [6] **HTTP POST Notification** = The Call Setup rule sends an HTTP POST request to notify an HTTP/S server of a specific condition and does not expect a response from the server. For example, you can configure a rule to notify the server of a 911 emergency call. To specify an HTTP server, use the 'Request Target' parameter (see below).
'Request Target' `request-target` [CallSetupRules_QueryTarget]	Defines one of the following, depending on the value configured for the 'Request Type' parameter (above). ■ **LDAP:** Defines an LDAP server (LDAP Server Group) on which to perform an LDAP query for a defined key. To configure LDAP Server Groups, see Configuring LDAP Server Groups. ■ **Dial Plan:** Defines a Dial Plan (name) in which to search for a defined key. To configure Dial Plans, see Configuring Dial Plans. ■ **ENUM:** Specifies the ENUM server on which to perform the ENUM query. The server is specified by IP Interface name (in the IP Interfaces table). The address of the ENUM server is the address of the 'Primary DNS' (and optionally, 'Secondary DNS') parameters that is configured for the specified IP Interface. If you don't specify an IP Interface or the specified IP Interface does not exist in the IP Interfaces table, the device uses the OAMP IP Interface. ■ **HTTP GET**, **HTTP POST Query**, and **HTTP POST Notification:** Defines the HTTP server to where the device sends the HTTP request. To configure HTTP servers, see Configuring Remote Web Services on page 316. To configure the key, use the 'Request Key' parameter (see below).
'Request Key' `request-key` [CallSetupRules_AttributesToQuery]	Defines the key to query. ■ For LDAP, the key string is queried on the LDAP server. ■ For Dial Plans, the key string is searched for in the specified Dial Plan. ■ For ENUM, the key string is queried on the ENUM server. ■ For HTTP GET and HTTP POST queries, the key string is queried on the HTTP server.

Parameter	Description

	The valid value is a string of up to 100 characters. Combined strings and values can be configured like in the Message Manipulations table, using the '+' operator. Single quotation marks (') can be used for specifying a constant string (e.g., '12345').
	You can use the built-in syntax editor to help you configure the field. Click the **Editor** button located alongside the field to open the Editor, and then simply follow the on-screen instructions.
	Examples:
	■ To LDAP query the AD attribute "mobile" that has the value of the destination user part of the incoming call:
	`'mobile=' + param.call.dst.user`
	■ To LDAP query the AD attribute "telephoneNumber" that has a redirect number:
	`'telephoneNumber=' + param.call.redirect + '*'`
	■ To query a Dial Plan for the source number:
	`param.call.src.user`
	■ To query an ENUM server for the URI of the called (destination) number:
	`param.call.dst.user`
	■ To send an HTTP POST to notify the HTTP server of call connection status:
	`'connectionStatus'`
	Note: The parameter is applicable only if the 'Request Type' parameter is configured to any value other than **None**.

'Attributes To Get' `attr-to-get` [CallSetupRules_ AttributesToGet]	Defines the Attributes of the queried LDAP record that the device must handle (e.g., retrieve value). The valid value is a string of up to 255 characters. Up to five attributes can be defined, each separated by a comma (e.g.,

Parameter	Description
	msRTCSIP-PrivateLine,msRTCSIP-Line,mobile). **Note:** ■ The parameter is applicable only if you configure the 'Request Type' parameter to **LDAP**. ■ The device saves the retrieved attributes' values for future use in other rules until the next LDAP query or until the call is connected. Thus, the device does not need to re-query the same attributes.
'Row Role' `row-role` [CallSetupRules_ RowRole]	Determines which condition must be met in order for this rule to be performed. ■ [0] **Use Current Condition** = The Condition configured for this rule must be matched in order to perform the configured action (default). ■ [1] **Use Previous Condition** = The Condition configured for the rule located directly above this rule in the Call Setup table must be matched in order to perform the configured action. This option lets you configure multiple actions for the same Condition.
'Condition' `condition` [CallSetupRules_ Condition]	Defines the condition that must exist for the device to perform the action. The valid value is a string of up to 200 characters (case-insensitive). Regular Expression (regex) can also be used. You can also use the built-in syntax editor to help you configure the field. Click the **Editor** button located alongside the field to open the Editor, and then simply follow the on-screen instructions. Examples: ■ LDAP: ✓ ldap.attr.mobile exists (if Attribute "mobile" exists in AD) ✓ param.call.dst.user == ldap.attr.msRTCSIP-PrivateLine (if called number is the same as the number in the Attribute "msRTCSIP-PrivateLine") ✓ ldap.found !exists (if LDAP record not found) ✓ ldap.err exists (if LDAP error exists) ■ Dial Plan: ✓ dialplan.found exists (if Dial Plan exists)

Parameter	Description
	✓ dialplan.found !exists (if Dial Plan queried key not found) ✓ dialplan.result=='uk' (if corresponding tag of the searched key is "uk") ■ ENUM: ✓ enum.found exists (if ENUM record of E.164 number exists) ■ HTTP GET or HTTP POST: • http.response.status == '200' (if the HTTP server responds with a 200 OK)
Action	
'Action Subject' `action-subject` [CallSetupRules_ ActionSubject]	Defines the element (e.g., SIP header, SIP parameter, SIP body, or Dial Plan tag) upon which you want to perform the action if the condition, configured in the 'Condition' parameter (see above) is met. The valid value is a string of up to 100 characters (case-insensitive). You can use the built-in syntax editor to help you configure the field. Click the **Editor** button located alongside the field to open the Editor, and then simply follow the on-screen instructions. Examples: ■ header.from contains '1234' ■ param.call.dst.user (called number) ■ param.call.src.user (calling number) ■ param.call.src.name (calling name) ■ param.call.redirect (redirect number) ■ param.call.src.host (source host) ■ param.call.dst.host (destination host) ■ srctags (source tag) ■ dsttags (destination tag) ■ header.content-type (for HTTP POST requests)
'Action Type' `action-type` [CallSetupRules_	Defines the type of action to perform. ■ [-1] **None** = No action is performed. This option is typically used for HTTP POST requests that are used for notifying the

Parameter	Description
ActionType]	HTTP server (e.g., when the 'Request Type' parameter is configured to **HTTP POST Notification**). If you configure the parameter to this option and it is the last rule in the table, the device processes the rule and then exits the table. If it is not the last rule, the device processes the rule and then checks the next rule. ■ [0] **Add** = (Default) Adds new message header, parameter or body elements. ■ [1] **Remove** = Removes message header, parameter, or body elements. ■ [2] **Modify** = Sets element to the new value (all element types). ■ [3] **Add Prefix** = Adds value at the beginning of the string (string element only). ■ [4] **Add Suffix** = Adds value at the end of the string (string element only). ■ [5] **Remove Suffix** = Removes value from the end of the string (string element only). ■ [6] **Remove Prefix** = Removes value from the beginning of the string (string element only). ■ [20] **Run Rules Set** = Performs a different Rule Set ID, specified in the 'Action Value' parameter (see below) ■ [21] **Exit** = Stops the Rule Set ID and returns a result ("true" or "false").
'Action Value' `action-value` [CallSetupRules_ ActionValue]	Defines a value that you want to use in the action. The valid value is a string of up to 300 characters (case-insensitive). You can use the built-in syntax editor to help you configure the field. Click the **Editor** button located alongside the field to open the Editor, and then simply follow the on-screen instructions. Examples: ■ '+9723976'+ldap.attr.alternateNumber ■ '9764000' ■ ldap.attr.displayName ■ enum.result.url

Parameter	Description
	■ srctags ■ http.response.body ■ application/x-www-form-urlencoded (for HTTP Content-Type header in HTTP requests) ■ **True** (if the 'Action Type' is configured to **Exit**) ■ **False** (if the 'Action Type' is configured to **Exit**)

Call Setup Rule Examples

Below are configuration examples for using Call Setup Rules.

- **Example 1:** This example configures the device to replace (manipulate) the incoming call's source number with a number retrieved from the AD by an LDAP query. The device queries the AD server for the attribute record, "telephoneNumber" whose value is the same as the received source number (e.g., "telephoneNumber =4064"). If such an Attribute exists, the device retrieves the number of the attribute record, "alternateNumber" and uses this number as the source number.
  - **Call Setup Rules table:**
    - 'Rules Set ID': **1**
    - 'Request Type': **LDAP**
    - 'Request Target': **LDAP-DC-CORP**
    - 'Request Key': **'telephoneNumber=' + param.call.src.user**
    - 'Attributes to Get': **alternateNumber**
    - 'Row Role': **Use Current Condition**
    - 'Condition': **ldap.attr. alternateNumber exists**
    - 'Action Subject': **param.call.src.user**
    - 'Action Type': **Modify**
    - 'Action Value': **ldap.attr. alternateNumber**
  - **IP-to-IP Routing table:** A single routing rule is assigned the Call Setup Rule Set ID.
    - (Index 1) 'Call Setup Rules Set ID': **1**
- **Example 2:** This example configures the device to replace (manipulate) the incoming call's calling name (caller ID) with a name retrieved from the AD by an LDAP query. The device queries the AD server for the attribute record, "telephoneNumber" whose value is the same as the received source number (e.g., "telephoneNumber =5098"). If such an attribute

is found, the device retrieves the name from the attribute record, "displayName" and uses this as the calling name in the incoming call.

- **Call Setup Rules table:**
  - 'Rules Set ID': **2**
  - 'Request Type': **LDAP**
  - 'Request Target': **LDAP-DC-CORP**
  - 'Request Key': **'telephoneNumber=' + param.call.src.user**
  - 'Attributes to Get': **displayName**
  - 'Row Role': **Use Current Condition**
  - 'Condition': **ldap.attr. displayName exists**
  - 'Action Subject': **param.call.src.name**
  - 'Action Type': **Modify**
  - 'Action Value': **ldap.attr. displayName**
- **IP-to-IP Routing table:** A single routing rule is assigned the Call Setup Rule Set ID.
  - (Index 1) 'Call Setup Rules Set ID': **2**

Example 3: This example configures the device to route the incoming call according to whether or not the source number of the incoming call also exists in the AD server. The device queries the AD server for the attribute record, "telephoneNumber" whose value is the same as the received source number (e.g., telephoneNumber=4064"). If such an attribute is found, the device sends the call to Skype for Business; if the query fails, the device sends the call to the PBX.

- **Call Setup Rules table:**
  - 'Rules Set ID': **3**
  - 'Request Type': **LDAP**
  - 'Request Target': **LDAP-DC-CORP**
  - 'Request Key': **'telephoneNumber=' + param.call.src.user**
  - 'Attributes to Get': **telephoneNumber**
  - 'Row Role': **Use Current Condition**
  - 'Condition': **ldap.found !exists**
  - 'Action Subject': -
  - 'Action Type': **Exit**
  - 'Action Value': **False**

  If the attribute record is found (i.e., condition is not met), the rule ends with a default exit result of true and uses the first routing rule (Skype for Business). If the attribute

record does not exist (i.e., condition is met), the rule exits with a "false" result and uses the second routing rule (PBX).

- **IP-to-IP Routing table:** Two routing rules are assigned with the same matching characteristics. Only the main routing rule is assigned a Call Setup Rules Set ID.
  - 'Index': 1
  - 'Call Setup Rules Set ID': **3**
  - 'Destination IP Group ID': **3** (IP Group for Skype for Business)
  - 'Index': 2
  - 'Destination IP Group ID': **4** (IP Group of PBX)

- **Example 4:** This example uses the msRTCSIP-DeploymentLocator AD attribute to determine if a user has migrated to Teams or not.
  - **Call Setup Rules table:**
    - 'Rules Set ID': **1**
    - 'Request Type': **LDAP**
    - 'Request Target': **LDAP-DC-CORP**
    - 'Request Key': **'(&(msRTCSIP-DeploymentLocator=SRV:)(msRTCSIP-Line=tel:'+param.call.dst.user+'*))'**
    - 'Attributes to Get': **msRTCSIP-DeploymentLocator**
    - 'Row Role': **Use Current Condition**
    - 'Condition': **ldap.attr.msRTCSIP-DeploymentLocator !exists**
    - 'Action Type': **Exit**
    - 'Action Value': **False**
  - **IP-to-IP Routing table:** A single routing rule is assigned the Call Setup Rule Set ID.
    - (Index 1) 'Call Setup Rules Set ID': **1**

- **Example 5:** This example enables routing based on LDAP queries and destination tags. The device queries the LDAP server for the attribute record "telephoneNumber" whose value is the destination number of the incoming call (e.g., "telephoneNumber=4064"). If the attribute-value combination is found, the device retrieves the string value of the attribute record "ofiSBCRouting" and creates a destination tag with the name of the retrieved string. The destination tag is then used as a matching characteristics in the IP-to-IP Routing table.
  - **Call Setup Rules table:**
    - 'Rules Set ID': **4**
    - 'Request Type': **LDAP**
    - 'Request Target': **LDAP-DC-CORP**
    - 'Request Key': **'telephoneNumber='+param.call.dst.user**

- 'Attributes to Get': **ofiSBCRouting**
- 'Row Role': **Use Current Condition**
- 'Condition': **ldap.found exists**
- 'Action Subject': **dsttags**
- 'Action Type': **Modify**
- 'Action Value': **ldap.attr.ofiSBCrouting**

- **IP Groups table:** 'Call Setup Rules Set ID': 4
- **IP-to-IP Routing table:**
  - 'Index': 1
  - 'Destination Tag': dep-sales
  - 'Destination IP Group': SALES
  - 'Index': 2
  - 'Destination Tag': dep-mkt
  - 'Destination IP Group': MKT
  - 'Index': 3
  - 'Destination Tag': dep-rd
  - 'Destination IP Group': RD

Example 6: This example configures the device to perform an ENUM query with an ENUM server to retrieve a SIP URI address for the called E.164 telephone number. The device then replaces (manipulates) the incoming call's E.164 destination number in the SIP Request-URI header with the URI retrieved from the ENUM server. The ENUM server's address is the address configured in the 'Primary DNS' parameter for the "ITSP-450" IP Interface in the IP Interfaces table.

- **Call Setup Rules table:**
  - 'Index': **0**
  - 'Rules Set ID': **4**
  - 'Request Type': **ENUM**
  - 'Request Target': **ITSP-450**
  - 'Request Key': **param.call.dst.user**
  - 'Condition': **enum.found exists**
  - 'Action Subject': **header.request-uri.url**
  - 'Action Type': **Modify**
  - 'Action Value': **enum.result.url**

- **IP Groups table:**
  - 'Call Setup Rules Set ID': **4**

- **Example 7:** For an example on HTTP GET operations, see Configuring an HTTP GET Web Service on page 334.
- **Example 8:** For an example on HTTP POST (notification) operations, see Configuring HTTP POST Web Service on page 336.

Configuring Dial Plans

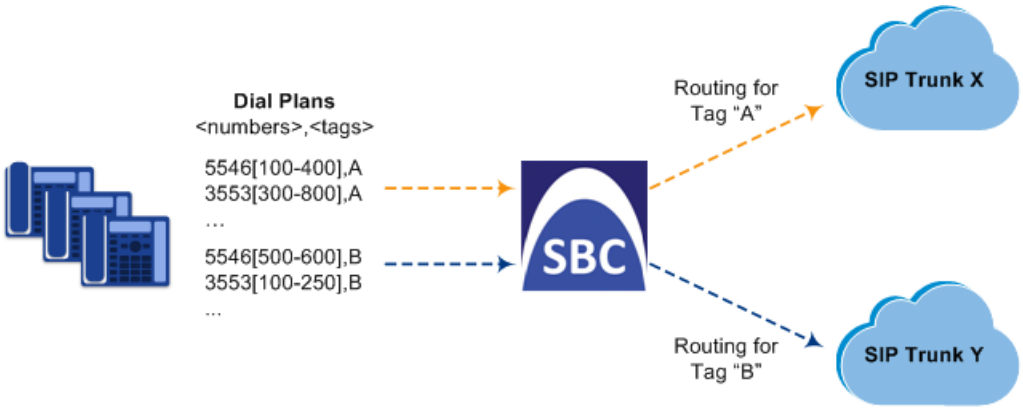
Dial Plans let you categorize incoming calls (source or destination) based on source or destination number. The device categorizes them by searching in the Dial Plan for rules that match these numbers according to prefix, suffix, or whole number. The categorization result in the Dial Plan is a *tag* corresponding to the matched rule. You can then use tags to represent these calls (source or destination) as matching characteristics (source or destination tags) for various configuration entities:

- SBC application:
  - IP-to-IP Routing rules (see Using Dial Plan Tags for IP-to-IP Routing)
  - Outbound Manipulations rules (Using Dial Plan Tags for Outbound Manipulation)
  - Call Setup Rules (Using Dial Plan Tags for Call Setup Rules)
  - Message Manipulation (Using Dial Plan Tags for Message Manipulation)

  You can assign a Dial Plan to an IP Group or SRD. After Classification and Inbound Manipulation, the device checks if a Dial Plan is associated with the incoming call. It first checks the source IP Group and if no Dial Plan is assigned, it checks the SRD. If a Dial Plan is assigned to the IP Group or SRD, the device first searches the Dial Plan for a dial plan rule that matches the source number and then it searches the Dial Plan for a rule that matches the destination number. If matching dial plan rules are found, the tags configured for these rules are used in the routing or manipulation processes as source or destination tags.

Note:
- User categorization by Dial Plan is done only after the device's Classification and Inbound Manipulation processes, and before the routing process.
- Once the device successfully categorizes an incoming call by Dial Plan, it not only uses the resultant tag in the immediate routing or manipulation process, but also in subsequent routing and manipulation processes that may occur, for example, due to alternative routing or local handling of call transfer and call forwarding (SIP 3xx\REFER).
- For manipulation, tags are applicable only to outbound manipulation.
- When tags are used in the IP-to-IP Routing table to determine destination IP Groups (i.e., 'Destination Type' parameter configured to **Destination Tag**), the device searches the Dial Plan for a matching **destination** (called) prefix number only.

The figure below shows a conceptual example of routing based on tags, where users categorized as tag "A" are routed to SIP Trunk "X" and those categorized as tag "B" are routed to SIP Trunk "Y":



The Dial Plan itself is a set of dial plan rules having the following attributes:

- **Prefix:** The prefix is matched against the source or destination number of the incoming call (e.g., SIP dialog-initiating request for IP calls).
- **Tag:** The tag corresponds to the matched prefix of the source or destination number and is the categorization result.

You can use various syntax notations to configure the prefix numbers in Dial Plan rules. You can configure the prefix as a complete number (all digits) or as a partial number using some digits and various syntax notations (patterns) to allow the device to match a Dial Plan rule for similar source or destination numbers. The device also employs a "best-match" method instead of a "first-match" method to match the source or destination numbers to the patterns configured in the Dial Plan. For more information, see the description of the 'Prefix' parameter (DialPlanRule_Prefix) described later in this section or see Notations and Priority Matching for Dial Plan Patterns on page 613.

> The maximum group of numbers (consisting of single numbers or range of numbers, or both) that can be configured for prefixes and suffixes for all the Dial Plan rules can be calculated by multiplying the maximum number of supported Dial Plan rules by six. For example, if the maximum number of Dial Plan rules is 100, then the maximum group of numbers is 600 (6*100). The following is an example of a Dial Plan rule that is configured with six groups of numbers (each separated by a comma), consisting of ranges and single numbers: [120-125,150,160-164,170,200,210-215]

Dial Plans are configured using two tables with "parent-child" relationship:

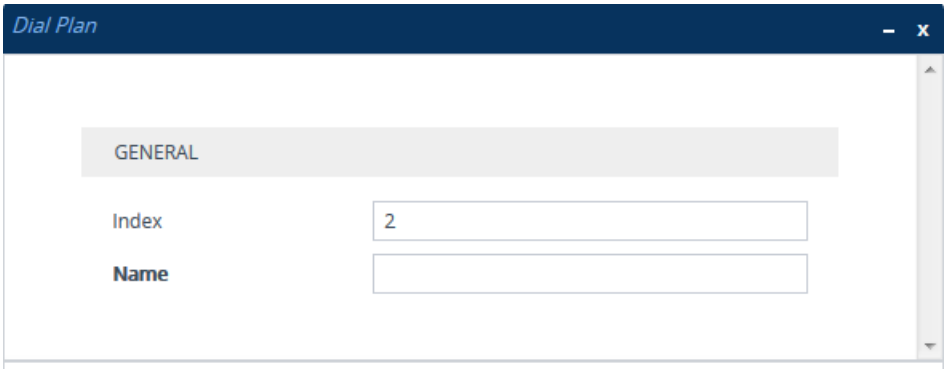
- Dial Plan table ("parent" table): Defines the name of the Dial Plan. You can configure up to 25 Dial Plans.
- Dial Plan Rule table ("child" table): Defines the actual dial plans (rules) per Dial Plan. You can configure up to 10,000 of Dial Plan rules in total (where all can be configured for one Dial Plan or configured between different Dial Plans).

The following procedure describes how to configure Dial Plans through the Web interface. You can also configure it through other management platforms:

- **Dial Plan table:** *ini* file [DialPlan] or CLI (`configure voip > sbc dial-plan`)
- **Dial Plan Rule table:** *ini* file (DialPlanRule) or CLI (configure voip > sbc dial-plan-rule)

➢ **To configure Dial Plans:**

1. Open the Dial Plan table (**Setup** menu > **Signaling & Media** tab > **SIP Definitions** folder > **Dial Plan**).
2. Click **New**; the following dialog box appears:

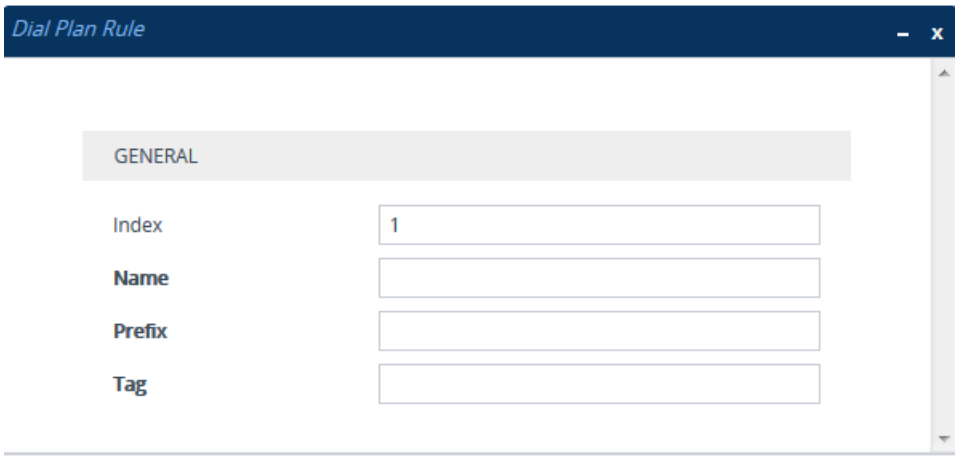


3. Configure a Dial Plan name according to the parameters described in the table below.
4. Click **Apply**.

Table 20-4: Dial Plan Table Parameter Descriptions

Parameter	Description
'Index' [DialPlan_Index]	Defines an index number for the new table row. **Note:** Each row must be configured with a unique index.
'Name' `name` [DialPlan_Name]	Defines a descriptive name, which is used when associating the row in other tables. The valid value is a string of up to 15 characters. **Note:** ■ Each row must be configured with a unique name. ■ The parameter value cannot contain a forward slash (/).

5. In the Dial Plan table, select the row for which you want to configure dial plan rules, and then click the **Dial Plan Rule** link located below the table; the Dial Plan Rule table appears.
6. Click **New**; the following dialog box appears:



7. Configure a dial plan rule according to the parameters described in the table below.
8. Click **New**, and then save your settings to flash memory.

Table 20-5: Dial Plan Rule Table Parameter Descriptions

Parameter	Description
'Index' index [DialPlanRule_RuleIndex]	Defines an index number for the new table row. **Note:** Each row must be configured with a unique index.
'Name' name [DialPlanRule_Name]	Defines a descriptive name, which is used when associating the row in other tables. The valid value is a string of up to 15 characters.
'Prefix' prefix [DialPlanRule_Prefix]	Defines the pattern to match the number (source or destination number) of the incoming call. The pattern can match the number based on prefix, suffix, or entire number. The valid value is a string of up to 50 characters. For valid notations and syntax, see Notations and Priority Matching for Dial Plan Patterns on the next page. **Note:** Dial Plan patterns are case-sensitive.
'Tag' tag [DialPlanRule_Tag]	Defines a tag(s). You must configure the tag with a name (e.g., "India") and optionally, with a value (i.e., name=value), for example, "Country=India", where "Country" is the tag's name and "India" is the tag's value. For guidelines on configuring tags, see the notes below. The valid value is a string of up to to 70 characters: ■ The tag's name can contain only the following characters: ✓ Alphanumeric characters (i.e., a-z, A-Z, and 0-9)

Parameter	Description
	✓ Special characters: • - (dash or hyphen) • ! (exclamation) • % (percentage) • * (asterisk) • _ (underscore) • ~ (tilde) • @ (at sign) ■ The tag's value can contain any character, except the following: ✓ = (equal) ✓ ; (semicolon) ✓ Spaces (including tab spaces) **Note:** ■ The tag is case-insensitive. ■ You can configure multiple tags, where each tag is separated by a semicolon, for example, "Belgium;Country1=England;Country2=India;Country3=10.1.1.1". ■ Tag names that have values must be unique. For example, "Country=England;Country=India" is an invalid configuration. An example of a valid configuration is "Country1=England;Country2=India". ■ You can configure only one tag without a value. For example, "Belgium;England" is an invalid configuration as both tag names don't have values. An example of a valid configuration is "Belgium;Country1=England;Country2=India" as only the tag name "Belgium" doesn't have a value. ■ You can configure the same tag in multiple Dial Plan rules. ■ In configuration tables that contain fields for assigning tags (e.g., IP-to-IP Routing table), if the field is left empty or configured with a single asterisk (*), any tag can match it.

Notations and Priority Matching for Dial Plan Patterns

The notations that you can use for configuring the 'Prefix field in the Dial Plan Rule table are described in the table below. As this field is used in the Dial Plan to match a number pattern

(source or destination) based on prefix, suffix or entire number, the notations are relevant to both prefix and suffix of the number (unless explicitly stated otherwise).

Notation	Description
0-9	Specific digit.
a-z	Lower-case letter. **Note:** Dial Plan matching is case-sensitive.
A-Z	Upper-case letter. **Note:** Dial Plan matching is case-sensitive.
x	Wildcard (metacharacter) that represents any single digit from 0 through 9. **Note:** ■ The wildcard is case-insensitive. ■ To represent the character "x", precede it with the escape "\" character. For example, to represent an upper-case "X", use this syntax: \X
z	Wildcard (metacharacter) that represents any single digit from 1 through 9. **Note:** ■ The wildcard is case-insensitive. ■ To represent the character "z", precede it with the escape "\" character. For example, to represent a lower-case "z", use this syntax: \z
n	Wildcard (metacharacter) that represents any single digit from 2 through 9. **Note:** ■ The wildcard is case-insensitive. ■ To represent the character "n", precede it with the escape "\" character. For example, to represent an upper-case "N", use this syntax: \N
.	(Dot) Wildcard (metacharacter) that represents any single character (letter, digit or symbol). To represent the dot "." character itself, precede it with the escape "\" character (see below).
*****	(Asterisk symbol) If it is the only character in the rule, it functions as a

Notation	Description
	wildcard (metacharacter) that represents any amount of digits or letters (i.e., matches everything). To represent the asterisk "*" symbol itself, precede it with the escape "\" character (see below). **Note:** You can't use a non-escaped * as part of the rule. For example, the following are invalid rules: "333*" or "192\.168\.0\.*"
****	(Backslash escape character) When it prefixes the wildcard character "n", "x", "z", or ".", the character is escaped and used literally instead of the wildcard function. For example, "10\.255\.255\.x" represents the IP address 10.255.255.[0-9]. As each dot (.) is prefixed by a backslash, the device considers these dots as the "." character (and not the . wildcard). In addition, as the "x" at the end of the value is not prefixed by a backslash, the device considers it the x wildcard.
#	(Pound or hash symbol) When used at the end of the prefix, it represents the end of the number. Examples: ■ 54324#: Represents the 5-digit number "54324". ■ 192\.168\.1\.[1-9]# and 192\.168\.1\.[01-96]#: Represent IP addresses 192.168.1.1 to 192.168.1.96
[n1-m1,n2-m2,a,b,c,...]	Represents a range of numbers for the prefix. The range can include both contiguous numbers and standalone numbers. Examples: ■ [123-130]: Represents a prefix number "123" through "130". ■ [123-130,455,766,780-790]: Represents a prefix number from "123" through "130", "455", "766", or "780" through 790". ■ [123,125,130]: Represents a prefix number "123", "125", or "130". **Note:** ■ The range (number ranges and single numbers) must contain the same amount of digits, as shown in the examples above where the number ranges and single numbers all contain three digits. ■ The device matches the numbers in the range and not the individual digits that make up the numbers. For example, if the rule's pattern is "[001-130]", the device matches strings such as "002", "012", "129" or "1001"; it doesn't match strings "2", "12", "301" or "0002". ■ You can't use an empty range (e.g., "+91[]").

Notation	Description
	■ Ranges can contain only digits (i.e., letters are not allowed). ■ The mixed notation can be configured with up to 19 digits, for example, "[123456789123456789,123456789123456791]". ■ The range (start and end) cannot be greater than 2,147,483,647, as in the example (which is invalid) "[20000000001-40000000001]".
([…])	Represents a range of numbers for the suffix. The range can include both contiguous numbers and standalone numbers. Examples: ■ ([123-130]): Represents a suffix number "123" through "130". ■ ([123-130,455,766,780-790]): Represents a suffix number from "123" through "130", "455", "766", or "780" through "790". ■ [123,125,130]: Represents a suffix number "123", "125", or "130". **Note:** ■ The range (number ranges and single numbers) must contain the same amount of digits, as shown in the examples above where the number ranges and single numbers all contain three digits. ■ The device matches the numbers in the range and not the individual digits that make up the numbers. For example, if the rule's pattern is "([001-130])", the device matches strings such as "002", "012", "129" or "9129"; it doesn't match strings "2", "12", "302" or "0200". ■ You can't use an empty suffix range (e.g., "+91([])"). ■ Ranges can contain only digits (i.e., letters are not allowed). ■ The mixed notation can be configured with up to 19 digits, for example, "([123456789123456789,123456789123456791])". ■ The range (start and end) cannot be greater than 2,147,483,647, as in the example (which is invalid) "([20000000001-40000000001])".
(…)	Represents a specific suffix, which can contain digits and letters. Examples: ■ [123-130](456): represent a number whose prefix number is "123" through "130" and whose suffix is "456". ■ 123(UK): represent a number whose prefix number is "123" and whose suffix is "UK". **Note:** You can't use an empty suffix (e.g., "+91()").

The device employs a "best-match" method instead of a "first-match" method to match the source/destination numbers to prefixes configured in the Dial Plan. The matching order is done digit-by-digit and from left to right.

The best match priority is listed below in chronological order:

1. Specific prefix
2. "x" wildcard, which denotes any digit (0 through 9)
3. Number range
4. "n" wildcard, which denotes a number from 2 through 9
5. "z" wildcard, which denotes a number from 1 through 9
6. Suffix, where the longest digits is first matched, for example, ([001-999]) takes precedence over ([01-99]) which takes precedence over ([1-9])
7. "." (dot), which denotes any single character

For example, the following table shows best matching priority for an incoming call with prefix number "5234":

Table 20-6: Dial Plan Best Match Priority

Dial Plan Prefix	Best Match Priority (Where 1 is Highest)
5234	1
523x	2
523[2-6]	3
523n	4
523z	5
523(4)	6
523.	7

When number ranges are used in Dial Plan rules (comma-separated standalone numbers or hyphenated range), best match priority is as follows:

- Dial Plan rules with ranges of **multiple standalone numbers**: The device chooses the matching rule in the Dial Plan Rule table that has the lowest row index number (i.e., listed higher up in the table). For example, if the prefix number of an incoming call is "110" and you have configured the below rules, the device chooses Index #0 because it has the lowest row index number (even though more numbers match the incoming call prefix number).

Index	Prefix
0	[1,3,5]
1	[110,120]

- Dial Plan rules with ranges of **contiguous numbers and the amount of possible matched numbers is identical**: The device chooses the matching rule in the Dial Plan Rule table that has the lowest row index number (i.e., listed higher up in the table). For example, if the prefix number of an incoming call is "110" and you have configured the below rules (each rule has a range of 3 possible matching numbers), the device chooses Index #0 because it has the lowest row index number (even though more numbers match the incoming call prefix number).

Index	Prefix
0	[1-3]
1	[10-12]

- Dial Plan rules with ranges of **contiguous number and the amount of possible matched numbers is different**: The device chooses the matching rule in the Dial Plan Rule table that has the least amount of numbers. For example, if the prefix number of an incoming call is "110" and you have configured the below rules (Index #0 with 7 possible matched numbers and Index #1 with 3 possible matched numbers), the device chooses Index #1 because it has less numbers.

Index	Prefix
0	[1-7]
1	[10-12]

- Dial Plan rules with ranges of **contiguous number** and **multiple standalone numbers**: The device chooses the matching rule in the Dial Plan Rule table that has the standalone number range (not contiguous range). For example, if the prefix number of an incoming call is "110" and you have configured the below rules (Index #0 is a standalone number range and Index #1 a contiguous range), the device chooses Index #0 because it is the standalone number range.

Index	Prefix
0	[1,2,3,4,5]
1	[1-3]

Additional examples of best match priority for Dial Plan rules configured with a specific number and optionally followed by the "x" notation or prefix or suffix range are shown below:

- For incoming calls with prefix number "5234", the rule with tag B is chosen (more specific for digit "4"):

Index	Prefix	Tag
0	523x	A
1	5234	B

- For incoming calls with prefix number "5234", the rule with tag A is chosen (see matching priority above):

Index	Prefix	Tag
0	523x	A
1	523[1-9]	B

- For incoming calls with prefix number "53211111", the rule with tag B is chosen (more specific for fourth digit):

Index	Prefix	Tag
0	532[1-9]1111	A
1	5321	B

- For incoming calls with prefix number "53124", the rule with tag B is chosen (more specific for digit "1"):

Index	Prefix	Tag
0	53([2-4])	A
1	531(4)	B

- For incoming calls with prefix number "321444", the rule with tag A is chosen and for incoming calls with prefix number "32144", the rule with tag B is chosen:

Index	Prefix	Tag
0	321xxx	A
1	321	B

- For incoming calls with prefix number "5324", the rule with tag B is chosen (prefix is more specific for digit "4"):

Index	Prefix	Tag
0	532[1-9]	A
1	532[2-4]	B

- For incoming calls with prefix number "53124", the rule with tag C is chosen (longest suffix - C has three digits, B two digits and A one digit):

Index	Prefix	Tag
0	53([2-4])	A
1	53([01-99])	B
2	53([001-999])	C

- For incoming calls with prefix number "53124", the rule with tag B is chosen (suffix is more specific for digit "4"):

Index	Prefix	Tag
0	53([2-4])	A
1	53(4)	B

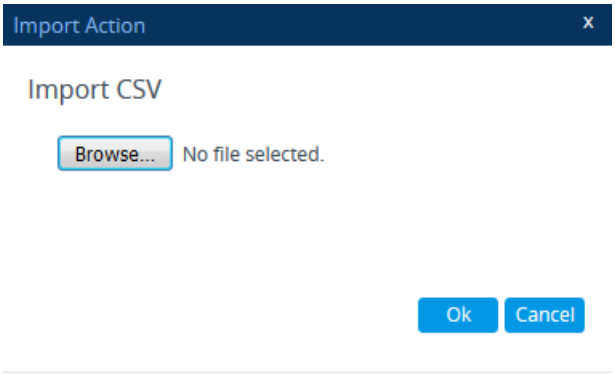
Importing Dial Plans

You can import Dial Plans and Dial Plan Rules from a comma-separated value (CSV) file on your local PC running the Web client.

- For creating Dial Plans in a CSV file for import, see Creating Dial Plan Files for Import.
- The CLI lets you import Dial Plans and Dial Plan rules from a file on a remote server, using the `import-csv-from` command under `(config-voip)# sbc dial-plan`. For more information, refer to the *CLI Reference Guide*.

➢ **To overwrite all existing Dial Plans with imported Dial Plan:**

1. Open the Dial Plan table.
2. From the 'Action' drop-down menu, choose **Import**; the following dialog box appears:

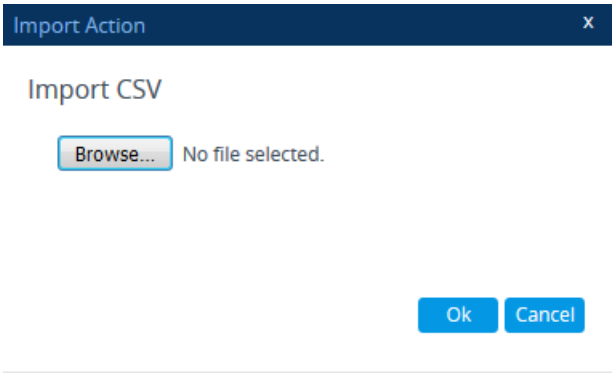


3. Use the **Browse** button to select the Dial Plan file on your PC, and then click **OK**.

- The file import feature only imports rules of Dial Plans that already exist in the Dial Plan table. If a Dial Plan in the file does not exist in the table, the specific Dial Plan is not imported.
- Make sure that the names of the Dial Plans in the imported file are **identical** to the existing Dial Plan names in the Dial Plan table; otherwise, Dial Plans in the file with different names are not imported.
- When importing a file, the rules in the imported file replace all existing rules of the corresponding Dial Plan. For existing Dial Plans in the Dial Plan table that are not listed in the imported file, the device deletes all their rules. For example, if the imported file contains only the Dial Plan "MyDialPlan1" and the device is currently configured with "MyDialPlan1" and "MyDialPlan2", the rules of "MyDialPlan1" in the imported file replace the rules of "MyDialPlan1" on the device, and the rules of "MyDialPlan2" on the device are deleted (the Dial Plan name itself remains).

➢ **To import Dial Plan rules for a specific Dial Plan:**

1. Open the Dial Plan table.
2. Select the required Dial Plan, and then click the **Dial Plan Rule** link; the Dial Plan Rule table opens, displaying all the rules of the selected Dial Plan.
3. From the 'Action' drop-down menu, choose **Import**; the following dialog box appears:



4. Use the **Browse** button to select the Dial Plan file on your PC, and then click **OK**.

The rules in the imported file replace **all** existing rules of the specific Dial Plan.

Creating Dial Plan Files

You can configure Dial Plans in an external file (*.csv) and then import them into the device, as described in Importing and Exporting Dial Plans. You can create the file using any text-based editor such as Notepad or Microsoft Excel. The file must be saved with the *.csv file name extension.

To configure Dial Plans in a file, use the following syntax:

```
DialPlanName,Name,Prefix,Tag
```

Where:

- *DialPlanName*: Name of the Dial Plan.
- *Name*: Name of the dial plan rule belonging to the Dial Plan.
- *Prefix:* Source or destination number prefix.
- *Tag:* Result of the user categorization and can be used as matching characteristics for routing and outbound manipulation

For example:

```
DialPlanName,Name,Prefix,Tag
PLAN1,rule_100,5511361xx,A
PLAN1,rule_101,551136184[4000-9999]#,B
MyDialPlan,My_rule_200,5511361840000#,itsp_1
MyDialPlan,My_rule_201,66666#,itsp_2
```

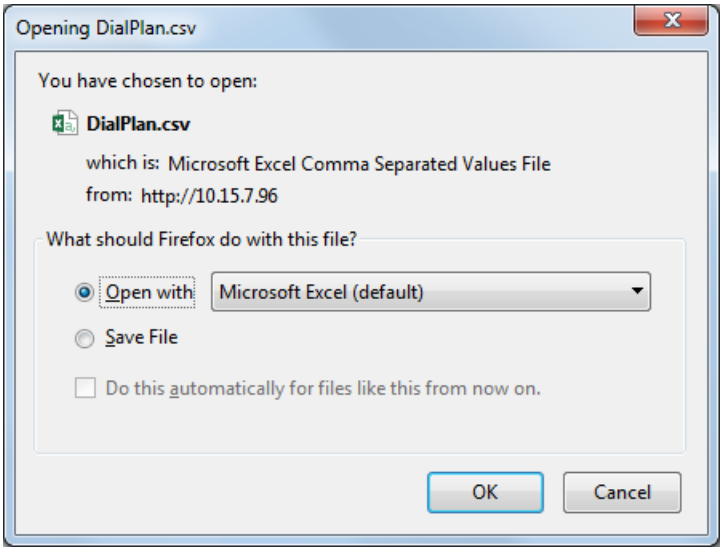
Exporting Dial Plans

You can export your configured Dial Plans in comma-separated value (CSV) file format to a folder on the local PC running the Web client.

The CLI lets you export Dial Plans and Dial Plan rules to a remote server, using the `export-csv-to` command under `(config-voip)# sbc dial-plan`. For more information, refer to the *CLI Reference Guide*.

➢ **To export all configured Dial Plans with their corresponding Dial Plan rules:**

1. Open the Dial Plan table.
2. From the 'Action' drop-down menu, choose **Export**; the following dialog box appears:



3. Select the **Save File** option, and then click **OK**; the file is saved to the default folder on your PC for downloading files.

➢ **To export Dial Plan rules of a specific Dial Plan:**

1. Open the Dial Plan table.
2. Select the required Dial Plan, and then click the **Dial Plan Rule** link; the Dial Plan Rule table opens, displaying the rules of the selected Dial Plan.
3. From the 'Action' drop-down menu, choose **Export**; a dialog box appears (as shown above).
4. Select the **Save File** option, and then click **OK**; the file is saved to the default folder on your PC for downloading files.

Using Dial Plan Tags for SBC IP-to-IP Routing

You can use Dial Plan tags with IP-to-IP Routing rules in the IP-to-IP Routing table, where tags can be used for the following:

- Matching routing rules by source and/or destination prefix numbers (see Using Dial Plan Tags for Matching Routing Rules)
- Locating destination IP Group (see Using Dial Plan Tags for Routing Destinations)

Using Dial Plan Tags for Matching Routing Rules

For deployments requiring hundreds of routing rules (which may exceed the maximum number of rules that can be configured in the IP-to-IP Routing table), you can employ tags to represent the many different calling (source URI user name) and called (destination URI user name) prefix numbers in your routing rules. Tags are typically implemented when you have users of many different called and/or calling numbers that need to be routed to the same destination (e.g., IP Group or IP address). In such a scenario, instead of configuring many routing rules to match all the required prefix numbers, you need only to configure a single routing rule using the tag to represent all the possible prefix numbers.

An example scenario where employing tags could be useful is in deployments where the device needs to service calls in a geographical area that consists of hundreds of local area codes, where each area code is serviced by one of two SIP Trunks in the network. In such a deployment, instead of configuring hundreds of routing rules to represent each local area code, you can simply configure two routing rules where each is assigned a unique tag representing a group of local area codes and the destination IP Group associated with the SIP Trunk servicing them.

- Source and destination tags can be used in the same routing rule.
- The same tag can be used for source and destination tags in the same routing rule.

➢ **To configure IP-to-IP routing based on tags:**

1. In the Dial Plan table, configure a Dial Plan (see Configuring Dial Plans). For example, the Dial Plan file below defines two tags, "LOC" and "INTL" to represent different called number prefixes for local and long distance (International) calls:

INDEX	NAME	PREFIX	TAG
0	Local	42520[3-5]	LOC
1	Local	425207	LOC
2	Local	42529	LOC
3	International	425200	INTL
4	International	425100	INTL

2. For the IP Group or SRD associated with the calls for which you want to use tag-based routing, assign the Dial Plan that you configured in Step 1.
   - IP Groups table: 'Dial Plan' parameter (IPGroup_SBCDialPlanName) - see Configuring IP Groups
   - SRDs table: 'Dial Plan' parameter (SRD_SBCDialPlanName) - see Configuring SRDs
3. In the IP-to-IP Routing table (see Configuring SBC IP-to-IP Routing Rules), configure a routing rule with the required destination and whose matching characteristics include the tag(s) that you configured in your Dial Plan in Step 1. The tags are assigned under the **Match** group, using the following parameters:
   - 'Source Tags' parameter (IP2IPRouting_SrcTags): tag denoting the calling user
   - 'Destination Tags' parameter (IP2IPRouting_DestTags): tag denoting the called user

Using Dial Plan Tags for Routing Destinations

You can use Dial Plan tags for determining the destination (IP Group) of an IP-to-IP Routing rule.

One of the benefits of using Dial Plan tags is that it can reduce the number of IP-to-IP Routing rules that you would normally need to configure. For example, assume that you need to route calls from IP Group "A" to two different IP Groups, "B" and "C", based on called (destination) prefix number (e.g., 102 and 103). When not using Dial Plan tags, you would need to configure

two IP-to-IP Routing rules, where one rule sends calls with prefix number 102 to IP Group "B" and another rule sends calls with prefix number 103 to IP Group "C". However, when using Dial Plan tags, you would need to configure only a single IP-to-IP Routing rule whose destination IP Group is based on a Dial Plan tag.

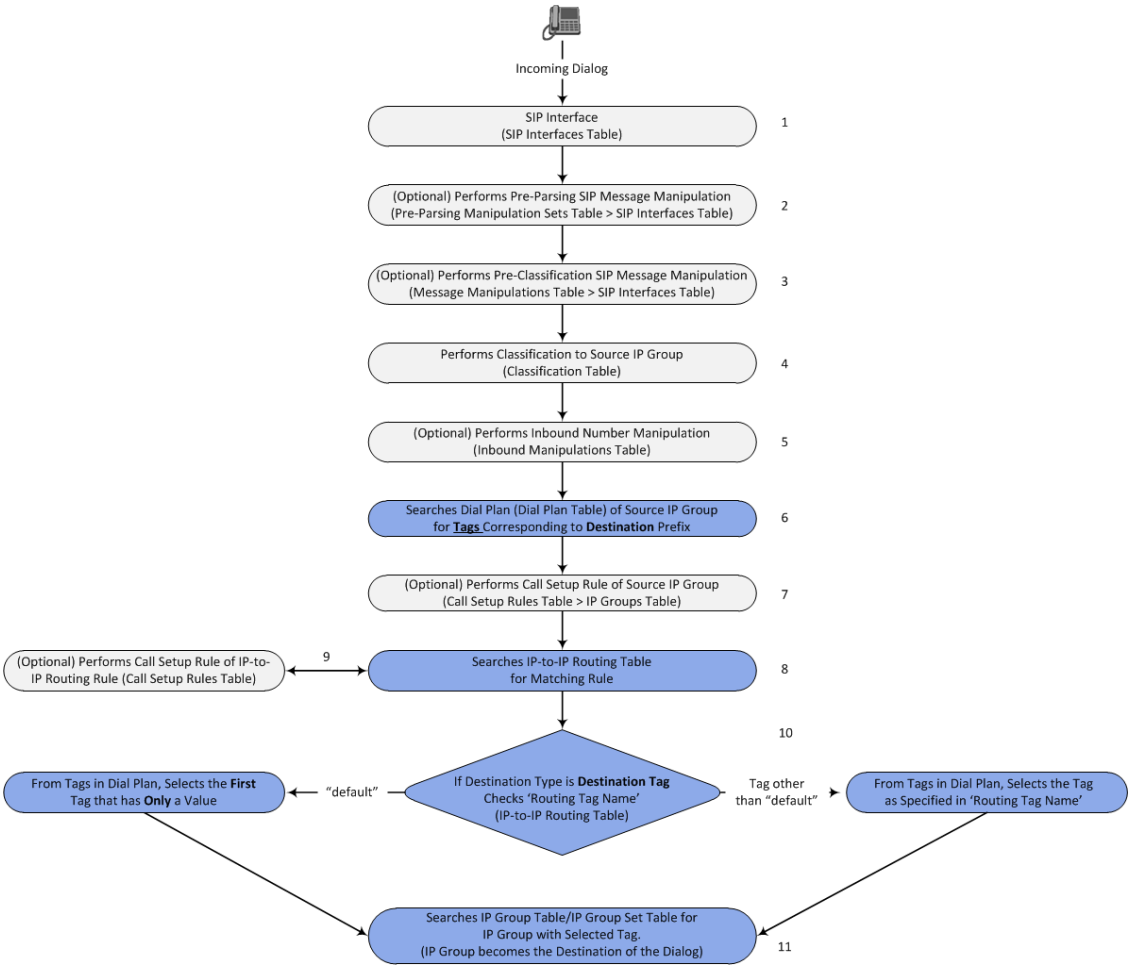
The following briefly describes the process for using Dial Plan tags in IP-to-IP Routing rules:

1. The device searches the Dial Plan Index, associated with the source IP Group of the incoming SIP dialog, for a Dial Plan rule whose 'Prefix' parameter is configured with the same called prefix number as the SIP dialog (e.g., 102). If found, the device inspects the tags in the 'Tag' parameter (e.g., "Country=England;City=London;Essex") configured for that Dial Plan rule.

> Instead of assigning a Dial Plan to the IP Group, you can assign a Call Setup rule ('Call Setup Rules Set ID' parameter) to determine the IP Group's tag. For more information on Call Setup rules (seeConfiguring Call Setup Rules on page 595).

2. The device searches for a matching rule in the IP-to-IP Routing table and if the 'Destination Type' parameter is configured to **Destination Tag**, it checks the tag name configured in the 'Routing Tag Name' parameter and compares it with the tags found in the Dial Plan rule. If the 'Routing Tag Name' parameter is configured as "default", the device selects the first tag name in the Dial Plan rule that is configured without a value, for example, "Essex" (see Step 1). If the 'Routing Tag Name' parameter is configured with a specific tag name (e.g., "Country"), the device selects the tag name with its value (e.g., "Country=England") in the Dial Plan rule.
3. The device searches the IP Groups table and IP Group Set table for an IP Group whose 'Tags' parameter is configured with the same tag as configured for the matching IP-to-IP Routing rule. If found, the device routes the call to this IP Group.

The following figure displays the device's SIP dialog processing when Dial Plan tags are used to determine the destination IP Group:



The following procedure describes how to configure routing to destination IP Groups determined by Dial Plan tags. The procedure is based on the following example scenario: Calls from IP Group "HQ" with destination (called) prefix number 102 are sent to IP Group "ENG" while calls with destination prefix number 103 are sent to IP Group "BEL". The destination IP Groups are determined by the Dial Plan tags, where the tag "Country=England" is used to send calls to IP Group "ENG" and the tag "Country=Belgium" is used to send calls to IP Group "BEL".

➢ To configure routing to destination IP Groups based on Dial Plan tags:

1. In the Dial Plan table, configure a Dial Plan with Dial Plan rules, where the 'Prefix' parameter is the **destination** (called) prefix number. In our example, we will configure a Dial Plan called "Dial Plan 1" with two Dial Plan rules:

Parameter	Index 0	Index 1
'Name'	UK	Bel-Neth
'Prefix'	102	103
'Tag'	Country=England;City=London	Holland;City=Amsterdam;Country=Belgium

The following displays the configuration in the Web interface of the Dial Plan rule for Index 0:

Regarding the 'Tag' parameter:

- Only one tag name **without** a value can be configured. In the above example, "Holland" is the tag name without a value. If an additional tag name is configured, for example, "Holland;France", the setting is invalid.
- Tag names with values (i.e., name=value) must be unique within a Dial Plan rule. In the above example, "Country=England" is a tag name with value. Configuring the parameter with "Country=England;Country=Scotland" is invalid. A valid configuration would be "Country=England;Country1=Scotland".

2. In the IP Groups table, configure your IP Groups. Make sure that you assign the source IP Group with the Dial Plan that you configured in Step 1 and that you configure each destination IP Group with one of the required Dial Plan tags. If the tag has a value, include it as well. In our example, we will configure three IP Groups:

Parameter	Index 0	Index 1	Index 2
'Name '	HQ	ENG	BEL
'Dial Plan'	Dial Plan 1	-	-
'Tags'	-	Country=England	Country=Belgium

3. In the IP-to-IP Routing table (see Configuring SBC IP-to-IP Routing on page 716), add a routing rule and configure the 'Destination Type' parameter to **Destination Tag** and the 'Routing Tag Name' to one of your Dial Plan tags. In our example, the tag "Country" is used:

Parameter	Index 0
'Name'	Europe
'Source IP Group'	HQ
'Destination Type'	**Destination Tag**
'Routing Tag Name'	Country

- If the IP-to-IP Routing rule (initial route) is also configured with a Call Setup rule ('Call Setup Rules Set ID' parameter) and it results in a different tag, and you have also configured alternative (or forking) IP-to-IP Routing rules with 'Destination Type' set to **Destination Tag**, then this new tag is used for the destinations of these alternative (or forking) rules, instead of the tag used for the initial route.
- Configure the 'Routing Tag Name' parameter with only the name of the tag (i.e., without the value, if exists). For example, instead of "Country=England", configure it as "Country" only.
- If the same Dial Plan tag is configured for an IP Group in the IP Groups table and an IP Group Set in the IP Group Set table, the IP Group Set takes precedence and the device sends the SIP dialog to the IP Group(s) belonging to the IP Group Set.

Dial Plan Backward Compatibility

This section is for backward compatibility **only**. It is recommended to migrate your Dial Plan configuration to the latest Dial Plan feature (see Using Dial Plan Tags for IP-to-IP Routing).

Configure prefix tags in the Dial Plan file using the following syntax:

```
[ PLAN<index> ]
<prefix number>,0,<prefix tag>
```

where:

- *Index* is the Dial Plan index
- *prefix number* is the called or calling number prefix (ranges can be defined in brackets)
- *prefix tag* is the user-defined prefix tag of up to nine characters, representing the prefix number

Each prefix tag type - called or calling - must be configured in a dedicated Dial Plan index number. For example, Dial Plan 1 can be for called prefix tags and Dial Plan 2 for calling prefix tags.

The example Dial Plan file below defines the prefix tags "LOCL"and "INTL" to represent different called number prefixes for local and long distance calls:

```
[ PLAN1 ]
42520[3-5],0,LOCL
425207,0,LOCL
42529,0,LOCL
425200,0,INTL
```

```
425100,0,INTL
....
```

- Called and calling prefix tags can be used in the same routing rule.
- When using prefix tags, you need to configure manipulation rules to remove the tags before the device sends the calls to their destinations.

The following procedure describes how to configure IP-to-IP routing using prefix tags.

➢ **To configure IP-to-IP routing using prefix tags:**

1. Configure a Dial Plan file with prefix tags, and then load the file to the device.
2. Add the prefix tags to the numbers of specific incoming calls using Inbound Manipulation rules:
   a. Open the Inbound Manipulations table (see Configuring IP-to-IP Inbound Manipulations), and then click **New**.
   b. Configure matching characteristics for the incoming call (e.g., set 'Source IP Group' to "1").
   c. From the 'Manipulated Item' drop-down list, select **Source** to add the tag to the calling URI user part, or **Destination** to add the tag to the called URI user part.
   d. Configure the Dial Plan index for which you configured your prefix tag, in the 'Prefix to Add' or 'Suffix to Add' fields, using the following syntax: $DialPlan<x>, where *x* is the Dial Plan index (0 to 7). For example, if the called number is 4252000555, the device manipulates it to LOCL4252000555.
3. Add an SBC IP-to-IP routing rule using the prefix tag to represent the different source or destination URI user parts:
   a. Open the IP-to-IP Routing table (see Configuring SBC IP-to-IP Routing Rules), and then click **New**.
   b. Configure the prefix tag in the 'Source Username Pattern' or 'Destination Username Pattern' fields (e.g., "LOCL", without the quotation marks).
   c. Continue configuring the rule as required.
4. Configure a manipulation rule to remove the prefix tags before the device sends the message to the destination:
   a. Open the Outbound Manipulations table (see Configuring IP-to-IP Outbound Manipulations), and then click **New**.
   b. Configure matching characteristics for the incoming call (e.g., set 'Source IP Group' to "1"), including calls with the prefix tag (in the 'Source Username Pattern' or 'Destination Username Pattern' fields, enter the prefix tag to remove).

c. Configure the 'Remove from Left' or 'Remove from Right' fields (depending on whether you added the tag at the beginning or end of the URI user part, respectively), enter the number of characters making up the tag.

Using Dial Plan Tags for SBC Outbound Manipulation

You can use Dial Plan tags to denote source and/or destination URI user names in Outbound Manipulation rules in the Outbound Manipulations table.

➢ **To configure Outbound Manipulation based on tags:**

1. In the Dial Plan table, configure a Dial Plan (see Configuring Dial Plans).
2. In the IP Group or SRD associated with the calls for which you want to use tag-based routing, assign the Dial Plan that you configured in Step 1.
   - IP Groups table: 'Dial Plan' parameter (IPGroup_SBCDialPlanName) - see Configuring IP Groups
   - SRDs table: 'Dial Plan' parameter (SRD_SBCDialPlanName) - see Configuring SRDs
3. In the Outbound Manipulations table (see Configuring IP-to-IP Outbound Manipulations), configure a rule with the required manipulation and whose matching characteristics include the tag(s) that you configured in your Dial Plan in Step 1. The tags are assigned using the following parameters:
   - 'Source Tags' parameter (IPOutboundManipulation_SrcTags): tag denoting the calling users
   - 'Destination Tags' parameter (IPOutboundManipulation_DestTags): tag denoting the called users

Using Dial Plan Tags for Call Setup Rules

You can use Dial Plan tags in Call Setup rules, configured in the Call Setup Rules table (see Configuring Call Setup Rules).

You can assign the Call Setup rule to an IP Group. The device runs the Call Setup rule for the source IP Group to which the incoming SIP dialog is classified, immediately before the routing process (i.e., Classification > Manipulation > Dial Plan table > Call Setup rules > Routing). The result of the Call Setup rule (i.e., source or destination tag) can be used in the IP-to-IP Routing table for any of the following:

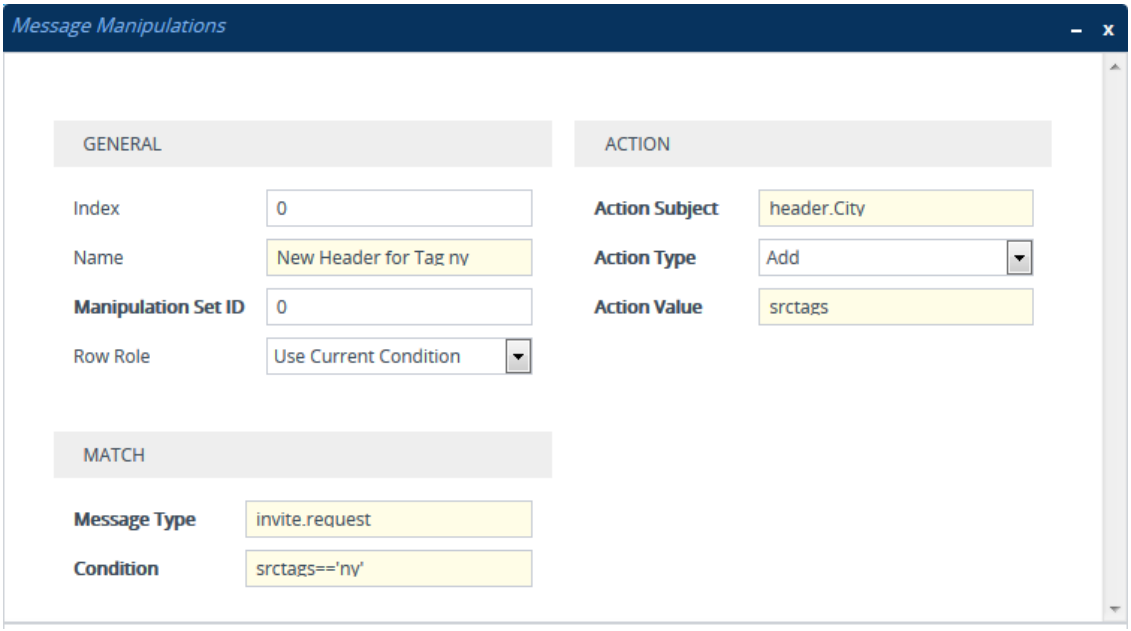
- As matching characteristics to find a suitable IP-to-IP Routing rule (initial route). To implement this, configure the rule's 'Source Tag' or 'Destination Tag' parameters to the resultant tag of the Call Setup rule.
- To determine the destination of the IP-to-IP Routing rule (initial route). To implement this, configure the rule's 'Destination Type' parameter to **Destination Tag** and the 'Routing Tag Name' parameter to the resultant tag of the Call Setup rule. The SIP dialog is sent to the IP Group that is configured with this same tag ('Tags' parameter in the IP Groups table).

When tags are used to determine the route's destination: If the IP-to-IP Routing rule (initial route) is also configured with a Call Setup rule ('Call Setup Rules Set ID' parameter) and it results in a different tag, and additional IP-to-IP Routing rules are configured as alternative routes ('Alternative Route Options' parameter), or the initial route is also configured with call forking ('Group Policy' is **Forking**), and the 'Destination Type' for these rules are configured to **Destination Tag**, then this new tag is used for the destination (instead of the tag used for the initial route).

You can configure Call Setup rules to query the Dial Plan table for a specified key (prefix) in a specified Dial Plan to obtain the corresponding tag. The Call Setup rule can then perform many different manipulations (based on Message Manipulation syntax), including modifying the name of the tag. The tags can be used only in the 'Condition', 'Action Subject' and 'Action Value' fields.

Using Dial Plan Tags for Message Manipulation

You can use Dial Plan tags (*srctags* and *dsttags*) in Message Manipulation rules, configured in the Message Manipulations table (see Configuring SIP Message Manipulation). The tags can be used only in the 'Condition' and 'Action Value' fields. For example, you can configure a rule that adds the SIP header "City" with the value "ny" (i.e., City: ny) to all outgoing SIP INVITE messages associated with the source tag "ny":



Dial Plan tags cannot be modified using Message Manipulation rules.

Configuring Push Notification Servers

The Push Notification Servers table lets you configure up to five Push Notification Servers. The device uses this table to determine which Push Notification Server to send push notification

requests for a specific user. The device searches the table for a row that is configured with the user's 'pn-provider' parameter value and if located, sends the push notification request to the Push Notification Server, using the address of the associated Remote Web Service.

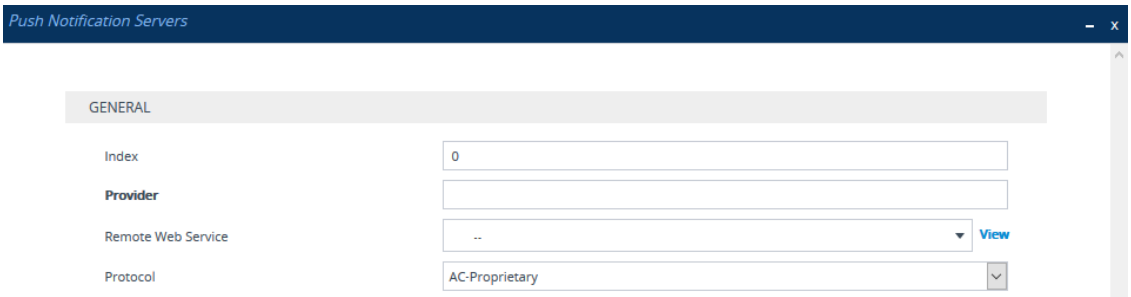
The Push Notification Service uses Push Notification Servers to send push notifications to "wake" end-user equipment (typically, mobile platforms) that have gone to "sleep" (e.g., to save resources such as battery life) so that they can receive traffic. The device can handle calls (and registration) for such SIP user agents (UAs), by interoperating with these third-party, Push Notification Servers (over HTTP, using RESTful APIs). For more information on Push Notification Service, see Configuring Push Notification Service on page 812.

Before you can configure a Push Notification Server in this table, you need to configure a Remote Web Service (HTTP-based server) to represent the Push Notification Server. The Remote Web Service defines the actual address (and other required parameters) of the server. You must configure the Remote Web Service with the 'Type' parameter set to **General**. To configure Remote Web Services, see Configuring Remote Web Services on page 316.

The following procedure describes how to configure Push Notification Servers through the Web interface. You can also configure it through ini file [PushNotificationServers] or CLI (`configure voip > sip-definition push-notification-servers`).

➢ To configure Push Notification Server:

1. Open the Push Notification Servers table (**Setup** menu > **Signaling & Media** tab > **SIP Definitions** folder > **Push Notification Servers**).
2. Click **New**; the following dialog box appears:



3. Configure a Push Notification Server according to the parameters described in the table below.
4. Click **Apply**.

Table 20-7: Push Notification Servers Table Parameter Descriptions

Parameter	Description
'Index' [PushNotificationServers_Index]	Defines an index number for the new table row. **Note:** Each row must be configured with a unique index.

Parameter	Description
'Provider' `provider` [PushNotificationServers_Provider]	Defines the type of the Push Notification Service. The type of the service provider is indicated in the SIP Contact header's 'pn-provider' parameter of the REGISTER message that is sent by the user to the device. For example, Android-based mobile phone platforms typically use Firebase Cloud Messaging (FCM) for its Push Notification Service. The value of the 'pn-provider' parameter for this service type is "fcm". Therefore, you would need to configure the 'Provider' parameter to "fcm" (without quotation marks). The valid value is a string of up to 10 characters. By default, no value is defined. To denote any provider, use the asterisk (*) wildcard character. **Note:** ■ You can configure this parameter to the * wildcard character for only one table row. ■ The parameter is mandatory.
'Remote Web Service' `remote-http-service` [PushNotificationServers_RemoteHTTPService]	Assigns a Remote Web Service, which defines the URL address (and other related parameters) of the HTTP-based Push Notification Server. To configure Remote Web Services, see Configuring Remote Web Services on page 316. **Note:** The parameter is mandatory.
'Protocol' `protocol` [PushNotificationServers_Protocol]	Defines the protocol for exchanging information between the device and the Push Notification Server. ■ [0] AC-Proprietary = (Default) The device exchanges information with the server using the JavaScript Object Notation (JSON) format.

21 SIP Message Manipulation

This section describes SIP message manipulation.

Configuring SIP Message Manipulation

The Message Manipulations table lets you configure up to 500 Message Manipulation rules. A Message Manipulation rule defines a manipulation sequence for SIP messages. SIP message manipulation enables the normalization of SIP messaging fields between communicating network segments. For example, it allows service providers to design their own policies on the SIP messaging fields that must be present before a SIP call enters their network. Similarly, enterprises and small businesses may have policies for the information that can enter or leave their networks for policy or security reasons from a service provider. SIP message manipulations can also be implemented to resolve incompatibilities between SIP devices inside the enterprise network.

Each Message Manipulation rule is configured with a Manipulation Set ID. You can create groups (sets) of Message Manipulation rules by assigning each of the relevant Message Manipulation rules to the same Manipulation Set ID. The Manipulation Set ID is then used to assign the rules to specific calls:

- Message manipulation rules can be applied pre- or post-classification:
  - Pre-classification Process: Message manipulation can be done on incoming SIP dialog-initiating messages (e.g., INVITE) prior to the classification process. You configure this by assigning the Manipulation Set ID to the SIP Interface on which the call is received (see Configuring SIP Interfaces).
  - Post-classification Process: Message manipulation can be done on inbound and/or outbound SIP messages after the call has been successfully classified. Manipulation occurs only after the routing process - inbound message manipulation is done first, then outbound number manipulation (see Configuring IP-to-IP Outbound Manipulations), and then outbound message manipulation. For viewing the call processing flow, see Call Processing of SIP Dialog Requests. You configure this by assigning the Manipulation Set ID to the relevant IP Group in the IP Groups table (see Configuring IP Groups).
- SIP requests initiated by the device: You can apply Message Manipulation rules to SIP requests that are initiated by the device, for example, SIP REGISTERs for certain entities (e.g., Accounts) and keep-alive by SIP OPTIONS. If the destination of the request is an IP Group, then the device uses the Inbound and Outbound Manipulation Sets that are assigned to the IP Group. If there is no IP Group for the destination or the IP Group is not assigned an Inbound or Outbound Manipulation Set, then the global parameters GWInboundManipulationSet or GWOutboundManipulationSet are used. The GWInboundManipulationSet parameter defines the Message Manipulation Set that is applied to incoming responses for requests that the device initiated. The

GWOutboundManipulationSet parameter defines the Message Manipulation Set that is applied to outgoing requests that the device initiates.

The device also supports a built-in SIP message normalization feature that can be enabled per Message Manipulation rule. The normalization feature removes unknown SIP message elements before forwarding the message. These elements can include SIP headers, SIP header parameters, and SDP body fields.

The SIP message manipulation feature supports the following:

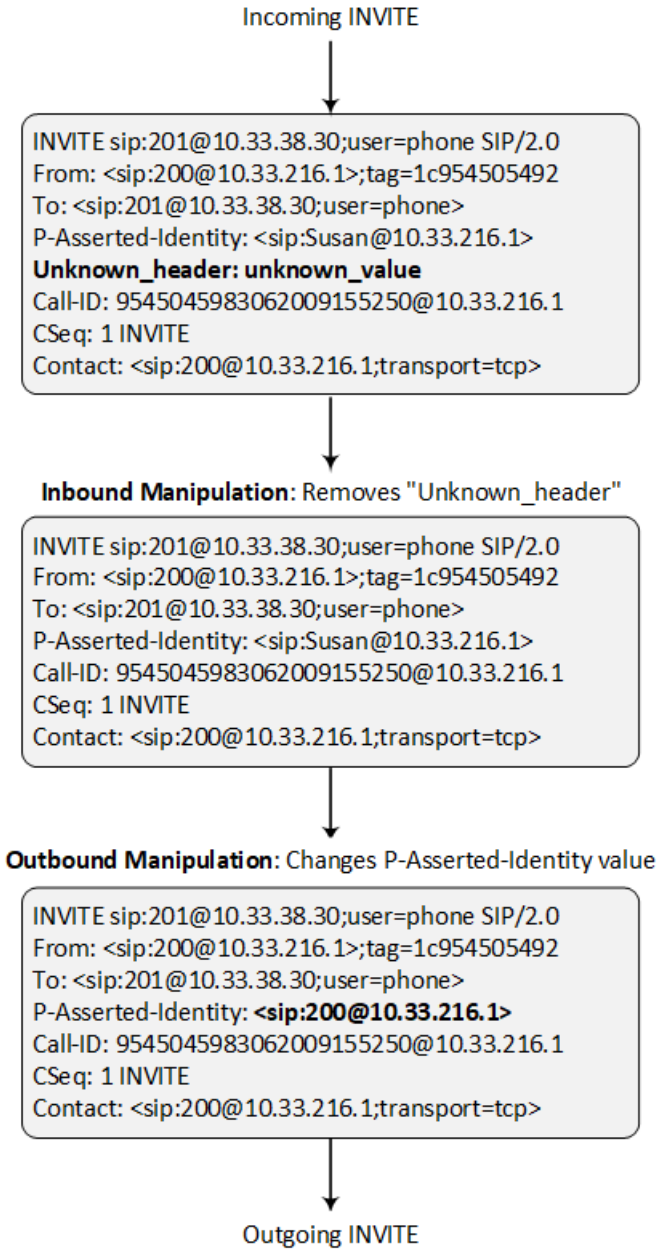
- Manipulation on SIP message type (Method, Request/Response, and Response type)
- Addition of new SIP headers
- Removal of SIP headers ("black list")
- Modification of SIP header components such as values, header values (e.g., URI value of the P-Asserted-Identity header can be copied to the From header), call's parameter values
- Deletion of SIP body (e.g., if a message body is not supported at the destination network this body is removed)
- Translating one SIP response code to another
- Topology hiding (generally present in SIP headers such as Via, Record Route, Route and Service-Route).
- Configurable identity hiding (information related to identity of subscribers, for example, P-Asserted-Identity, Referred-By, Identity and Identity-Info)
- Multiple manipulation rules on the same SIP message
- Apply conditions per rule - the condition can be on parts of the message or call's parameters
- Apply Message Manipulation Set twice on SIP REGISTER messages -- first on the initial incoming unauthenticated REGISTER, and then again on the incoming authenticated SIP message received after the device sends a SIP 401 response for challenging the initial REGISTER request. For more information and for enabling this feature, see the [AuthenticatedMessageHandling] parameter.
- Multiple manipulation rules using the same condition. The following figure shows a configuration example where Rules #1 and #2 ('Row Rule' configured to **Use Previous Condition**) use the same condition as configured for Rule #0 ('Row Rule' configured to **Use Current Condition**). For more information, see the description of the 'Row Rule' parameter in this section.

INDEX	NAME	MANIPULATION SET ID	MESSAGE TYPE	CONDITION	ACTION SUBJECT	ACTION TYPE	ACTION VALUE	ROW ROLE
0	To Header Urgent	1	Invite.Request	Header.Request-URI.UR	Header.To	Modify	Header.To + ';urgent=1'	Use Current Condition
1	Add Emergency	1			Header.Priority	Add	'emergency'	Use Previous Condition
2	User-Agent	1			Header.User-Agent	Modify	'trunk-a'	Use Previous Condition

The following figure illustrates an example of a SIP message that is manipulated by the device as follows:

1. Removes the "Unknown_header: unknown_value" in the incoming message.

2. Changes the P-Asserted-Identity header value to "sip:200@10.33.216.1" in the outgoing message.



This manipulation example is done by configuring two Message Manipulation rules, where Rule #1 is assigned to the source IP Group and Rule #2 to the destination IP Group.

Parameter	Rule 1	Rule 2
Message Type	Invite.request	Invite.request
Condition	Header.Unkown_header !contains 'unknown_value'	Header.P-Asserted-Identity.URL.User == 'Susan'

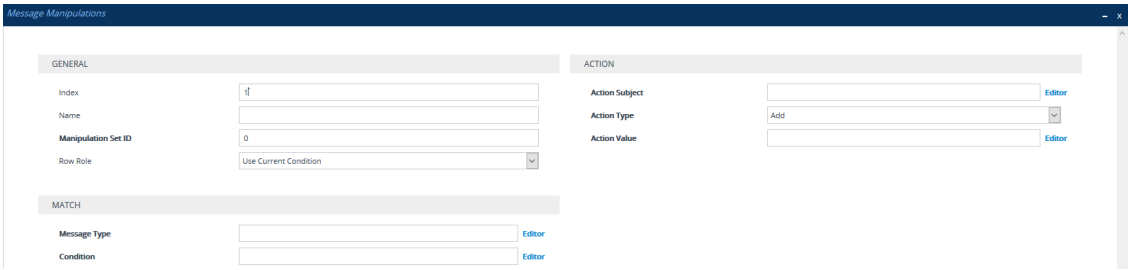
Parameter	Rule 1	Rule 2
Action Subject	Header.Unkown_header	Header.P-Asserted-Identity
Action Type	Remove	Modify
Action Value		'<sip:200@212.3.216.1>'

- For a detailed description of the syntax used for configuring Message Manipulation rules, refer to the *SIP Message Manipulation Syntax Reference Guide*.
- Inbound message manipulation is done only after the Classification, inbound and outbound number manipulation, and routing processes.
- Each message can be manipulated twice - on the source leg and on the destination leg (i.e., source and destination IP Groups).
- Unknown SIP parts can only be added or removed.
- SIP manipulations do not allow you to remove or add mandatory SIP headers. They can only be modified and only on requests that initiate new dialogs. Mandatory SIP headers include To, From, Via, CSeq, Call-Id, and Max-Forwards.
- The SIP Group Name (IPGroup_SIPGroupName) parameter overrides inbound message manipulation rules that manipulate the host name in Request-URI, To, and/or From SIP headers. If you configure a SIP Group Name for the IP Group (see Configuring IP Groups) and you want to manipulate the host name in these SIP headers, you must apply your manipulation rule (Manipulation Set ID) to the IP Group as an Outbound Message Manipulation Set (IPGroup_ OutboundManSet), when the IP Group is the destination of the call. If you apply the Manipulation Set as an Inbound Message Manipulation Set (IPGroup_ InboundManSet), when the IP Group is the source of the call, the manipulation rule will be overridden by the SIP Group Name.

The following procedure describes how to configure Message Manipulation rules through the Web interface. You can also configure it through ini file [MessageManipulations] or CLI (`configure voip > message message-manipulations`).

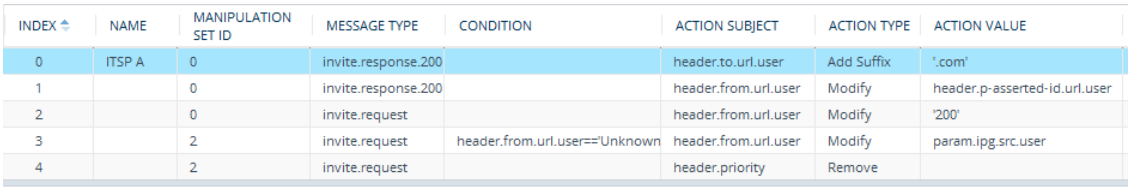
➢ To configure SIP message manipulation rules:

1. Open the Message Manipulations table (**Setup** menu > **Signaling & Media** tab > **Message Manipulation** folder > **Message Manipulations**).
2. Click **New**; the following dialog box appears:



3. Configure a Message Manipulation rule according to the parameters described in the table below.
4. Click **Apply**.

An example of configured message manipulation rules are shown in the figure below:



INDEX	NAME	MANIPULATION SET ID	MESSAGE TYPE	CONDITION	ACTION SUBJECT	ACTION TYPE	ACTION VALUE
0	ITSP A	0	invite.response.200		header.to.url.user	Add Suffix	'.com'
1		0	invite.response.200		header.from.url.user	Modify	header.p-asserted-id.url.user
2		0	invite.request		header.from.url.user	Modify	'200'
3		2	invite.request	header.from.url.user=='Unknown'	header.from.url.user	Modify	param.ipg.src.user
4		2	invite.request		header.priority	Remove	

- **Index 0:** Adds the suffix ".com" to the host part of the To header.
- **Index 1:** Changes the user part of the From header to the user part of the P-Asserted-ID.
- **Index 2:** Changes the user part of the SIP From header to "200".
- **Index 3:** If the user part of the From header equals "unknown", then it is changed according to the srcIPGroup call's parameter.
- **Index 4:** Removes the Priority header from an incoming INVITE message.

Table 21-1: Message Manipulations Parameter Descriptions

Parameter	Description
General	
'Index' [MessageManipulations_Index]	Defines an index number for the new table row. **Note:** Each row must be configured with a unique index.
'Name' `manipulation-name` [MessageManipulations_ ManipulationName]	Defines a descriptive name, which is used when associating the row in other tables. The valid value is a string of up to 40 characters.
'Manipulation Set ID' `manipulation-set-id` [MessageManipulations_ManSetID]	Defines a Manipulation Set ID for the rule. You can define the same Manipulation Set ID for multiple rules to create a group of rules. The Manipulation Set ID is used to assign the manipulation rules to an IP Group (in the IP Groups table) for inbound and/or outbound messages. The valid value is 0 to 19. The default is 0.
'Row Role' `row-role`	Determines which message manipulation condition (configured by the 'Condition' parameter) to use for the rule.

Parameter	Description
[MessageManipulations_RowRole]	■ [0] **Use Current Condition** = (Default) The condition configured in the table row of the rule is used. ■ [1] **Use Previous Condition** = The condition configured in the first table row above the rule that is configured to **Use Current Condition** is used. For example, if Index 3 is configured to **Use Current Condition** and Index 4 and 5 are configured to **Use Previous Condition**, Index 4 and 5 use the condition configured for Index 3. A configuration example is shown in the beginning of this section. The option allows you to use the same condition for multiple manipulation rules. **Note:** ■ When configured to **Use Previous Condition**, the 'Message Type' and 'Condition' parameters are not applicable and if configured are ignored. ■ When multiple manipulation rules apply to the same header, the next rule applies to the resultant string of the previous rule.
Match	
'Message Type' `message-type` [MessageManipulations_ MessageType]	Defines the SIP message type that you want to manipulate. The valid value is a string (case-insensitive) denoting the SIP message. You can use the built-in syntax editor to help you configure the field. Click the **Editor** button located alongside the field to open the Editor, and then simply follow the on-screen instructions. For example: ■ Empty = rule applies to all messages ■ Invite = rule applies to all INVITE requests and responses ■ Invite.Request = rule applies to INVITE requests

Parameter	Description
	■ Invite.Response = rule applies to INVITE responses ■ subscribe.response.2xx = rule applies to SUBSCRIBE confirmation responses **Note:** Currently, SIP 100 Trying messages cannot be manipulated.
'Condition' `condition` [MessageManipulations_Condition]	Defines the condition that must exist for the rule to be applied. The valid value is a string (case-insensitive). You can use the built-in syntax editor to help you configure the field. Click the **Editor** button located alongside the field to open the Editor, and then simply follow the on-screen instructions. For example: ■ header.from.url.user== '100' (indicates that the user part of the From header must have the value "100") ■ header.contact.param.expires > '3600' ■ header.to.url.host contains 'domain' ■ param.call.dst.user != '100'
Action	
'Action Subject' `action-subject` [MessageManipulations_ ActionSubject]	Defines the SIP header upon which the manipulation is performed. The valid value is a string (case-insensitive). You can use the built-in syntax editor to help you configure the field. Click the **Editor** button located alongside the field to open the Editor, and then simply follow the on-screen instructions.
'Action Type' `action-type` [MessageManipulations_ActionType]	Defines the type of manipulation. ■ [0] **Add** = (Default) Adds new header/param/body (header or parameter elements). ■ [1] **Remove** = Removes header/param/body (header or parameter elements).

Parameter	Description
	■ [2] **Modify** = Sets element to the new value (all element types). ■ [3] **Add Prefix** = Adds value at the beginning of the string (string element only). ■ [4] **Add Suffix** = Adds value at the end of the string (string element only). ■ [5] **Remove Suffix** = Removes value from the end of the string (string element only). ■ [6] **Remove Prefix** = Removes value from the beginning of the string (string element only). ■ [7] **Normalize** = Removes unknown SIP message elements before forwarding the message.
'Action Value' `action-value` [MessageManipulations_ActionValue]	Defines a value that you want to use in the manipulation. The default value is a string (case-insensitive) in the following syntax: ■ string/<message-element>/<call-param> + ■ string/<message-element>/<call-param> For example: ■ 'itsp.com' ■ header.from.url.user ■ param.call.dst.user ■ param.call.dst.host + '.com' ■ param.call.src.user + '<' + header.from.url.user + '@' + header.p-asserted-id.url.host + '>' ■ Func.To-Upper(Param.Call.Src.Host) You can use the built-in syntax editor to help you configure the field. Click the **Editor** button located alongside the field to open the Editor, and then simply follow the on-screen instructions. **Note:** Only single quotation marks must be used.

Configuring Message Condition Rules

The Message Conditions table lets you configure up to 500 Message Condition rules. A Message Condition defines special conditions (requisites) for incoming SIP messages. These rules can be used as additional matching criteria for the following:

- Classification rules in the Classification table (see Configuring Classification Rules)
- IP-to-IP routing rules in the IP-to-IP Routing table (see Configuring SBC IP-to-IP Routing Rules)
- Outbound Manipulation rules in the Outbound Manipulations table (see Configuring IP-to-IP Outbound Manipulations)

Message Condition rules are configured using the same syntax as that used for Conditions when configuring Message Manipulation rules in the Message Manipulations table (see Configuring SIP Message Manipulation). You can configure simple Message Condition rules, for example, "header.to.host contains company", meaning SIP messages whose To header has a host part containing the string "company". You can configure complex rules using the "AND" or "OR" Boolean operands and also use regular expressions (regex), for example:

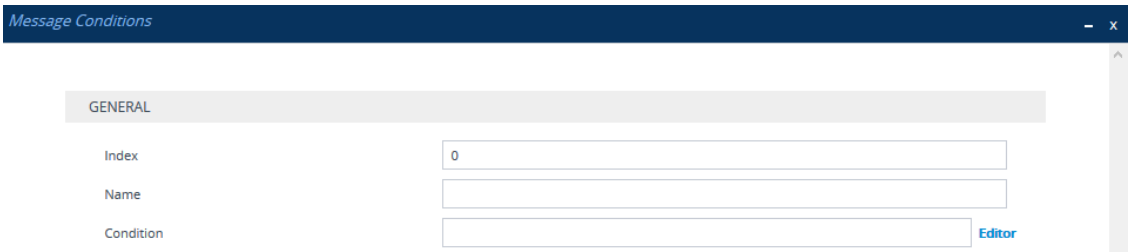
- "body.sdp regex pcmu" can be used to enable routing based on the offered codec (G.711 Mu) in the incoming SDP message.
- "body.sdp regex (AVP[0-9|\|\s]*\s8[\s|\|\n])" can be used to enable routing based on payload type 8 in the incoming SDP message.

> For a description on SIP message manipulation syntax, refer to the *Syntax for SIP Message Manipulation Reference Guide*.

The following procedure describes how to configure Message Condition rules through the Web interface. You can also configure it through ini file [ConditionTable] or CLI (`configure voip > sbc routing condition-table`).

➢ To configure a Message Condition rule:

1. Open the Message Conditions table (**Setup** menu > **Signaling & Media** tab > **Message Manipulation** folder > **Message Conditions**).
2. Click **New**; the following dialog box appears:



3. Configure a Message Condition rule according to the parameters described in the table below.
4. Click **Apply**.

An example of configured Message Condition rules is shown in the figure below:

INDEX	NAME	CONDITION
0	IP Group user	param.ipg.src.type==user
1	Contains SIP Via Header	header.via.exists
2	"101" user part in From header	header.from.url.user=='101'

- **Index 0:** Incoming SIP dialog that is classified as belonging to a User-type IP Group.
- **Index 1:** Incoming SIP dialog that contains a SIP Via header.
- **Index 2:** Incoming SIP dialog with "101" as the user part in the SIP From header.

Table 21-2: Message Conditions Table Parameter Descriptions

Parameter	Description
'Index' [ConditionTable_Index]	Defines an index number for the new table row. **Note:** Each row must be configured with a unique index.
'Name' `name` [ConditionTable_Name]	Defines a descriptive name, which is used when associating the row in other tables. The valid value is a string of up to 59 characters. **Note:** The parameter value cannot contain a forward slash (/).
'Condition' `condition` [ConditionTable_Condition]	Defines the condition of the SIP message. The valid value is a string. You can use the built-in syntax editor to help you configure the field. Click the **Editor** button located alongside the field to open the Editor, and then simply follow the on-screen instructions. **Note:** User and host parts must be enclosed by a single quotation mark ('...').

Configuring SIP Message Policy Rules

The Message Policies table lets you configure up to 20 SIP Message Policy rules. SIP Message Policy rules are used to block (blacklist) unwanted incoming SIP messages or permit (whitelist) receipt of desired SIP messages. You can configure legal and illegal characteristics of SIP messages. This feature is helpful against VoIP fuzzing (also known as robustness testing), which sends different types of packets to its "victims" for finding bugs and vulnerabilities. For example, the attacker might try sending a SIP message containing either an oversized parameter or too many occurrences of a parameter.

You can also enable the Message Policy to protect the device against incoming SIP messages with malicious signature patterns, which identify specific scanning tools used by attackers to search for SIP servers in a network. To configure Malicious Signatures, see Configuring Malicious Signatures.

Each Message Policy rule can be configured with the following:

- Maximum message length
- Maximum header length
- Maximum message body length
- Maximum number of headers
- Maximum number of bodies
- Option to send 400 "Bad Request" response if message request is rejected
- Blacklist and whitelist for defined methods (e.g., INVITE)
- Blacklist and whitelist for defined bodies
- Malicious Signatures

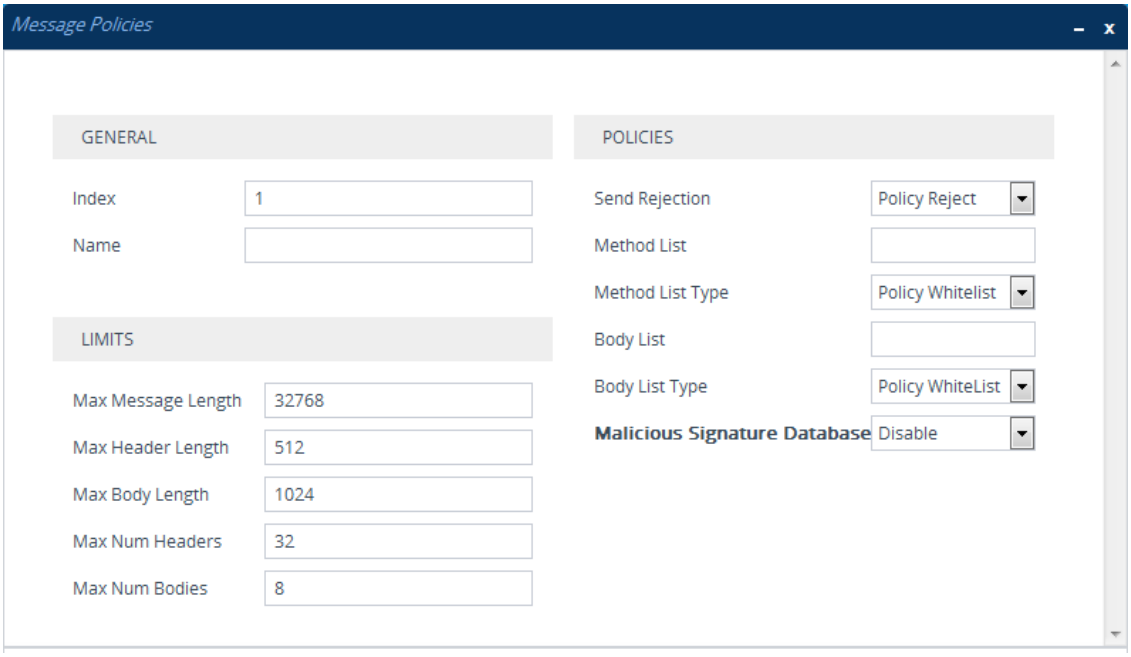
The Message Policies table provides a default Message Policy called "Malicious Signature DB Protection" (Index 0), which is based only on Malicious Signatures and discards SIP messages identified with any of the signature patterns configured in the Malicious Signature table.

To apply a SIP Message Policy rule to calls, you need to assign it to the SIP Interface associated with the relevant IP Group (see Configuring SIP Interfaces).

The following procedure describes how to configure Message Policy rules through the Web interface. You can also configure it through ini file [MessagePolicy] or CLI (`configure voip > message message-policy`).

➢ **To configure SIP Message Policy rules:**

1. Open the Message Policies table (**Setup** menu > **Signaling & Media** tab > **Message Manipulation** folder > **Message Policies**).
2. Click **New**; the following dialog box appears:



3. Configure a Message Policy rule according to the parameters described in the table below.
4. Click **Apply**.

Table 21-3: Message Policies Table Parameter Descriptions

Parameter	Description
General	
'Index' [MessagePolicy_Index]	Defines an index number for the new table row. **Note:** Each row must be configured with a unique index.
'Name' `name` [MessagePolicy_Name]	Defines a descriptive name, which is used when associating the row in other tables. The valid value is a string of up to 40 characters. **Note:** ■ Each row must be configured with a unique name. ■ The parameter value cannot contain a forward slash (/).
Limits	
'Max Message Length' `max-message-length` [MessagePolicy_ MaxMessageLength]	Defines the maximum SIP message length. The valid value is up to 32,768 characters. The default is 32,768.
'Max Header Length'	Defines the maximum SIP header length.

Parameter	Description
max-header-length [MessagePolicy_ MaxHeaderLength]	The valid value is up to 512 characters. The default is 512.
'Max Body Length' max-body-length [MessagePolicy_ MaxBodyLength]	Defines the maximum SIP message body length. This is the value of the Content-Length header. The valid value is up to 1,024 characters. The default is 1,024.
'Max Num Headers' max-num-headers [MessagePolicy_ MaxNumHeaders]	Defines the maximum number of SIP headers. The valid value is any number up to 32. The default is 32. **Note:** The device supports up to 20 SIP Record-Route headers that can be received in a SIP INVITE request or a 200 OK response. If it receives more than this, it responds with a SIP 513 'Message Too Large' response.
'Max Num Bodies' max-num-bodies [MessagePolicy_ MaxNumBodies]	Defines the maximum number of bodies (e.g., SDP) in the SIP message. The valid value is any number up to 8. The default is 8.
Policies	
'Send Rejection' send-rejection [MessagePolicy_ SendRejection]	Defines whether the device sends a SIP response if it rejects a message request due to the Message Policy. The default response code is SIP 400 "Bad Request". To configure a different response code, use the MessagePolicyRejectResponseType parameter. ■ [0] **Policy Reject** = (Default) The device discards the message and sends a SIP response to reject the request. ■ [1] **Policy Drop** = The device discards the message without sending any response.
SIP Method Blacklist-Whitelist Policy	
'Method List' method-list [MessagePolicy_MethodList]	Defines SIP methods (e.g., INVITE\BYE) to blacklist or whitelist. Multiple methods are separated by a backslash (\). The method values are case-insensitive.
'Method List Type'	Defines the policy (blacklist or whitelist) for the SIP

Parameter	Description
`method-list-type` [MessagePolicy_ MethodListType]	methods specified in the 'Method List' parameter (above). ■ [0] **Policy Blacklist** = The specified methods are rejected. ■ [1] **Policy Whitelist** = (Default) Only the specified methods are allowed; the others are rejected.
SIP Body Blacklist-Whitelist Policy	
'Body List' `body-list` [MessagePolicy_BodyList]	Defines the SIP body type (i.e., value of the Content-Type header) to blacklist or whitelist. For example, application/sdp. The values of the parameter are case-sensitive.
'Body List Type' `body-list-type` [MessagePolicy_ BodyListType]	Defines the policy (blacklist or whitelist) for the SIP body specified in the 'Body List' parameter (above). ■ [0] **Policy Blacklist** =The specified SIP body is rejected. ■ [1] **Policy Whitelist** = (Default) Only the specified SIP body is allowed; the others are rejected.
Malicious Signature	
'Malicious Signature Database' `signature-db-enable` [MessagePolicy_ UseMaliciousSignatureDB]	Enables the use of the Malicious Signature database (signature-based detection). ■ [0] **Disable** (default) ■ [1] **Enable** To configure Malicious Signatures, see Configuring Malicious Signatures.

Configuring Pre-Parsing Manipulation Rules

The Pre-Parsing Manipulation Set table lets you configure up to 10 Pre-Parsing Manipulation Sets. Pre-Parsing Manipulation allows you to manipulate incoming SIP messages (dialog-initiating and in-dialog) before they are parsed (as an object) by the device. In other words, messages can be manipulated in their original format (plain text) as received from the network. Pre-Parsing Manipulation may be useful, for example, to overcome parser strictness or to "allow" possible parsing errors.

To use a configured Pre-Parsing Manipulation Set, you need to assign it to a SIP Interface (see Configuring SIP Interfaces). The device performs Pre-Parsing Manipulation before Pre-Classification Manipulation and Classification.

Pre-Parsing Manipulation rules are defined by the SIP message element to manipulate (for example, INVITE), the pattern based on regular expression (regex) to search for (match) in the incoming message, and the regex pattern to replace the matched pattern.

> For a detailed description of supported regex syntax, refer to the *Syntax for SIP Message Manipulation Reference Guide*

Pre-Parsing Manipulation is configured using two tables with "parent-child" relationship:

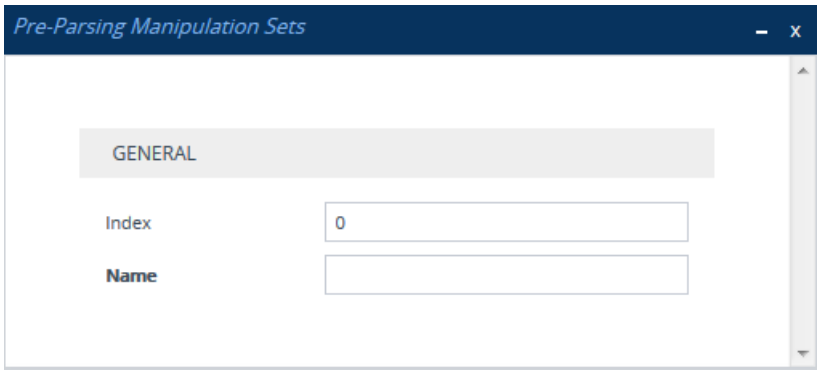
- Pre-Parsing Manipulation Sets table ("parent"): Defines a descriptive name for the Pre-Parsing Manipulation Set.
- Pre-Parsing Manipulation Rules table ("child"): Defines the actual manipulation rule. You can configure up to 10 rules per Pre-Parsing Manipulation Set.

The following procedure describes how to configure Pre-Parsing Manipulation Sets through the Web interface. You can also configure it through other management platforms:

- **Pre-Parsing Manipulation Sets table:** *ini* file [PreParsingManipulationSets] or CLI (`configure voip > message pre-parsing-manip-sets`)
- **Pre-Parsing Manipulation Rules table:** *ini* file [PreParsingManipulationRules] or CLI (`configure voip > message pre-parsing-manip-rules`)

➢ To configure Pre-Parsing Manipulation Sets:

1. Open the Pre-Parsing Manipulation Sets table (**Setup** menu > **Signaling & Media** tab > **Message Manipulation** folder > **Pre-Parsing Manipulation Sets**).
2. Click **New**; the following dialog box appears:

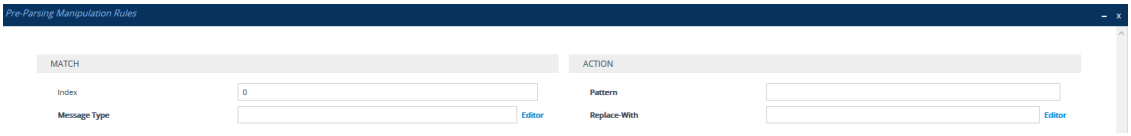


3. Configure a Pre-Parsing Manipulation Set name according to the parameters described in the table below.
4. Click **Apply**.

Table 21-4: Pre-Parsing Manipulation Set Table Parameter Descriptions

Parameter	Description
'Index' [PreParsingManipulationSets_Index]	Defines an index number for the new table row. **Note:** Each row must be configured with a unique index.
'Name' `name` [PreParsingManipulationSets_Name]	Defines a descriptive name, which is used when associating the row in other tables. The valid value is a string of up to 40 characters. **Note:** ■ Each row must be configured with a unique name. ■ The parameter value cannot contain a forward slash (/).

5. In the Pre-Parsing Manipulation Sets table, select the row, and then click the **Pre-Parsing Manipulation Rules** link located below the table; the Pre-Parsing Manipulation Rules table appears.
6. Click **New**; the following dialog box appears:



7. Configure a rule according to the parameters described in the table below.
8. Click **New**, and then save your settings to flash memory.

Table 21-5: Pre-Parsing Manipulation Rules Table Parameter Descriptions

Parameter	Description
Match	
'Index' [PreParsingManipulationRules_RuleIndex]	Defines an index number for the new table row. **Note:** Each row must be configured with a unique index.
'Message Type' `message-type` [PreParsingManipulationRules_MessageType]	Defines the SIP message type to which you want to apply the rule. The following syntax is supported: ■ To apply the rule to any message type, leave the field empty or configure it to **any**. ■ SIP requests:

Parameter	Description
	✓ **any.request:** The rule is applied to any request. ✓ **<SIP Method>.request:** The rule is applied to the specified SIP Method (e.g., invite.request). ■ SIP responses: ✓ **any.response:** The rule is applied to any response. ✓ **response.<response code>:** The rule is applied to messages with the specified response (e.g., response.200 for SIP 200 or response.1xx for any provisional response). You can use the built-in syntax editor to help you configure the field. Click the **Editor** button located alongside the field to open the Editor, and then simply follow the on-screen instructions.
Action	
'Pattern' `pattern` [PreParsingManipulationRules_Pattern]	Defines a pattern, based on regex, to search for (match) in the incoming message. For more information on regex, refer to the *Syntax for SIP Message Manipulation Reference Guide*.
'Replace-With' `replace-with` [PreParsingManipulationRules_ReplaceWith]	Defines a pattern, based on regex, to replace the matched pattern (defined above). You can use the built-in syntax editor to help you configure the field. Click the **Editor** button located alongside the field to open the Editor, and then simply follow the on-screen instructions. For more information on regex, refer to the *Syntax for SIP Message Manipulation Reference Guide*.

Part V

Session Border Controller Application

22 SBC Overview

This section provides an overview of the device's SBC application.

- For guidelines on how to deploy your SBC device, refer to the *SBC Design Guide* document.
- The SBC feature is available only if the device is installed with a License Key that includes this feature. For installing a License Key, see License Key.
- For the maximum number of supported SBC sessions, and SBC users than can be registered in the device's registration database, see Technical Specifications.

Feature List

The SBC application supports the following main features:

- NAT traversal: The device supports NAT traversal, allowing, for example, communication with ITSPs with globally unique IP addresses and with far-end users located behind NAT on the WAN. The device supports this by:
  - Continually registering far-end users with its users registration database.
  - Maintaining remote NAT binding state by frequent registrations and thereby, off-loading far-end registrations from the LAN IP PBX.
  - Using Symmetric RTP (RFC 4961) to overcome bearer NAT traversal.
- VoIP firewall and security for signaling and media:
  - SIP signaling:
    - Deep and stateful inspection of all SIP signaling packets.
    - SIP dialog initiations may be rejected based on values of incoming SIP INVITE message and other Layer-3 characteristics.
    - Packets not belonging to an authorized SIP dialog are discarded.
  - RTP:
    - Opening pinholes (ports) in the device's firewall based on SDP offer-answer negotiations.
    - Deep packet inspection of all RTP packets.
    - Late rogue detection - if a SIP session was gracefully terminated and someone tries to "ride on it" with rogue traffic from the already terminated RTP and SIP context, the VoIP Firewall prevents this from occurring.
    - Disconnects call (after user-defined time) if RTP connection is broken.
    - Black/White lists for both Layer-3 firewall and SIP classification.

- Stateful Proxy Operation Mode: The device can act as a Stateful Proxy by enabling SIP messages to traverse it transparently (with minimal interference) between the inbound and outbound legs.
- B2BUA and Topology Hiding: The device intrinsically supports topology hiding, limiting the amount of topology information displayed to external parties. For example, IP addresses of ITSPs' equipment (e.g. proxies, gateways, and application servers) can be hidden from outside parties. The device's topology hiding is provided by implementing back-to-back user agent (B2BUA) leg routing:
  - Strips all incoming SIP Via header fields and creates a new Via value for the outgoing message.
  - Each leg has its own Route/Record Route set.
  - User-defined manipulation of SIP To, From, and Request-URI host names.
  - Generates a new SIP Call-ID header value (different between legs).
  - Changes the SIP Contact header and sets it to the device's address.
  - Layer-3 topology hiding by modifying source IP address in the SIP IP header.
- SIP normalization: The device supports SIP normalization, whereby the SBC application can overcome interoperability problems between SIP user agents. This is achieved by the following:
  - Manipulation of SIP URI user and host parts.
  - Connection to ITSP SIP trunks on behalf of an IP-PBX - the device can register and utilize user and password to authenticate for the IP-PBX.
- Survivability:
  - Routing calls to alternative routes such as the PSTN.
  - Routing calls between user agents in the local network using a dynamic database (built according to registrations of SIP user agents).
- Routing:
  - IP-to-IP routing translations of SIP, UDP, TCP, TLS (when extensive transcoding is not required).
  - Load balancing and redundancy of SIP servers.
  - Routing according to Request-URI\Specific IP address\Proxy\FQDN.
  - Alternative routing.
  - Routing between different Layer-3 networks (e.g., LAN and WAN).
- Load balancing\redundancy of SIP servers.
- ITSP accounts.
- SIP URI user and host name manipulations.

- Coder transcoding.

B2BUA and Stateful Proxy Operating Modes

The device can operate in one or both of the following SBC modes:

- **Back-to-Back User Agent (B2BUA):** Maintains independent sessions toward the endpoints, processing an incoming request as a user agent server (UAS) on the inbound leg, and processing the outgoing request as a user agent client (UAC) on the outbound leg. SIP messages are modified regarding headers between the legs and all the device's interworking features may be applied.
- **Stateful Proxy Server:** SIP messages traverse the device transparently (with minimal interference) between the inbound and outbound legs, for connecting SIP endpoints.

By default, the device's B2BUA mode changes SIP dialog identifiers and topology data in SIP messages traversing through it:

- Call identifiers: Replaces the From-header tag and Call-ID header so that they are different for each leg (inbound and outbound).
- Routing headers:
  - Removes all Via headers in incoming requests and sends the outgoing message with its own Via header.
  - Doesn't forward any Record-Route headers from the inbound to outbound leg, and vice versa.
  - Replaces the address of the Contact header in the incoming message with its own address in the outgoing message.
- Replaces the User-Agent/ Server header value in the outgoing message, and replaces the original value with itself in the incoming message.

In contrast, when the device operates in Stateful Proxy mode, the device by default forwards SIP messages transparently (unchanged) between SIP endpoints (from inbound to outbound legs). The device retains the SIP dialog identifiers and topology headers received in the incoming message and sends them as is in the outgoing message. The device handles the above mentioned headers transparently (i.e., they remain unchanged) or according to configuration (enabling partial transparency), and only adds itself as the top-most Via header and optionally, to the Record-Route list. To configure the handling of these headers for partial transparency, use the following IP Profile parameters (see Configuring IP Profiles):

- IpProfile_SBCRemoteRepresentationMode: Contact and Record-Route headers
- IpProfile_SBCKeepVIAHeaders: Via headers
- IpProfile_SBCKeepUserAgentHeader: User-Agent headers
- IpProfile_SBCKeepRoutingHeaders: Record-Route headers
- IpProfile_SBCRemoteMultipleEarlyDialogs: To-header tags

Thus, the Stateful Proxy mode provides full SIP transparency (no topology hiding) or asymmetric topology hiding. Below is an example of a SIP dialog-initiating request when operating in Stateful Proxy mode for full transparency, showing all the incoming SIP headers retained in the outgoing INVITE message.

Incoming INVITE

```
INVITE sip:bob@domain.com SIP/2.0
To: Bob <sip:bob@domain.com>
From: Alice
<sip:alice@caller.com>;tag=100
Call-ID: callid1@caller.com
Contact: <sip:alice@pc1.caller.com>
Via: SIP/2.0/UDP pc2.com;branch=brancn2
Via: SIP/2.0/UDP pc1.com;branch=brancn1
Record-Route: <pc2.com;lr>
Record-Route: <pc1.com;lr>
CSeq: 666 INVITE
User-Agent: IPPv3.1
Max-Forwards: 70
Content-Type: application/sdp
Content-Length: 142

v=0
...
```

Outgoing INVITE

```
INVITE sip:bob@domain.com SIP/2.0
To: Bob <sip:bob@domain.com>
From: Alice
<sip:alice@caller.com>;tag=100
Call-ID: callid1@caller.com
Contact: <sip:alice@pc1.caller.com>
Via: SIP/2.0/UDP Proxy-IP;branch=brancn3
Via: SIP/2.0/UDP pc2.com;branch=brancn2
Via: SIP/2.0/UDP pc1.com;branch=brancn1
Record-Route: <Proxy-IP;lr>
Record-Route: <pc2.com;lr>
Record-Route: <pc1.com;lr>
CSeq: 666 INVITE
User-Agent: IPPv3.1
Max-Forwards: 70
Content-Type: application/sdp
Content-Length: 142
v=0
...
```

Some of the reasons for implementing Stateful Proxy mode include:

- B2BUA typically hides certain SIP headers for topology hiding. In specific setups, some SIP servers require the inclusion of these headers to know the history of the SIP request. In such setups, the requirement may be asymmetric topology hiding, whereby SIP traffic toward the SIP server must expose these headers whereas SIP traffic toward the users must not expose these headers.
- B2BUA changes the call identifiers between the inbound and outbound SBC legs and therefore, call parties may indicate call identifiers that are not relayed to the other leg. Some SIP functionalities are achieved by conveying the SIP call identifiers either in SIP specific headers (e.g., Replaces) or in the message bodies (e.g. Dialog Info in an XML body).
- In some setups, the SIP client authenticates using a hash that is performed on one or more of the headers that B2BUA changes (removes). Therefore, implementing B2BUA would cause authentication to fail.
- For facilitating debugging procedures, some administrators require that the value in the Call-ID header remains unchanged between the inbound and outbound SBC legs. As B2BUA changes the Call-ID header, such debugging requirements would fail.

The operating mode can be configured per the following configuration entities:

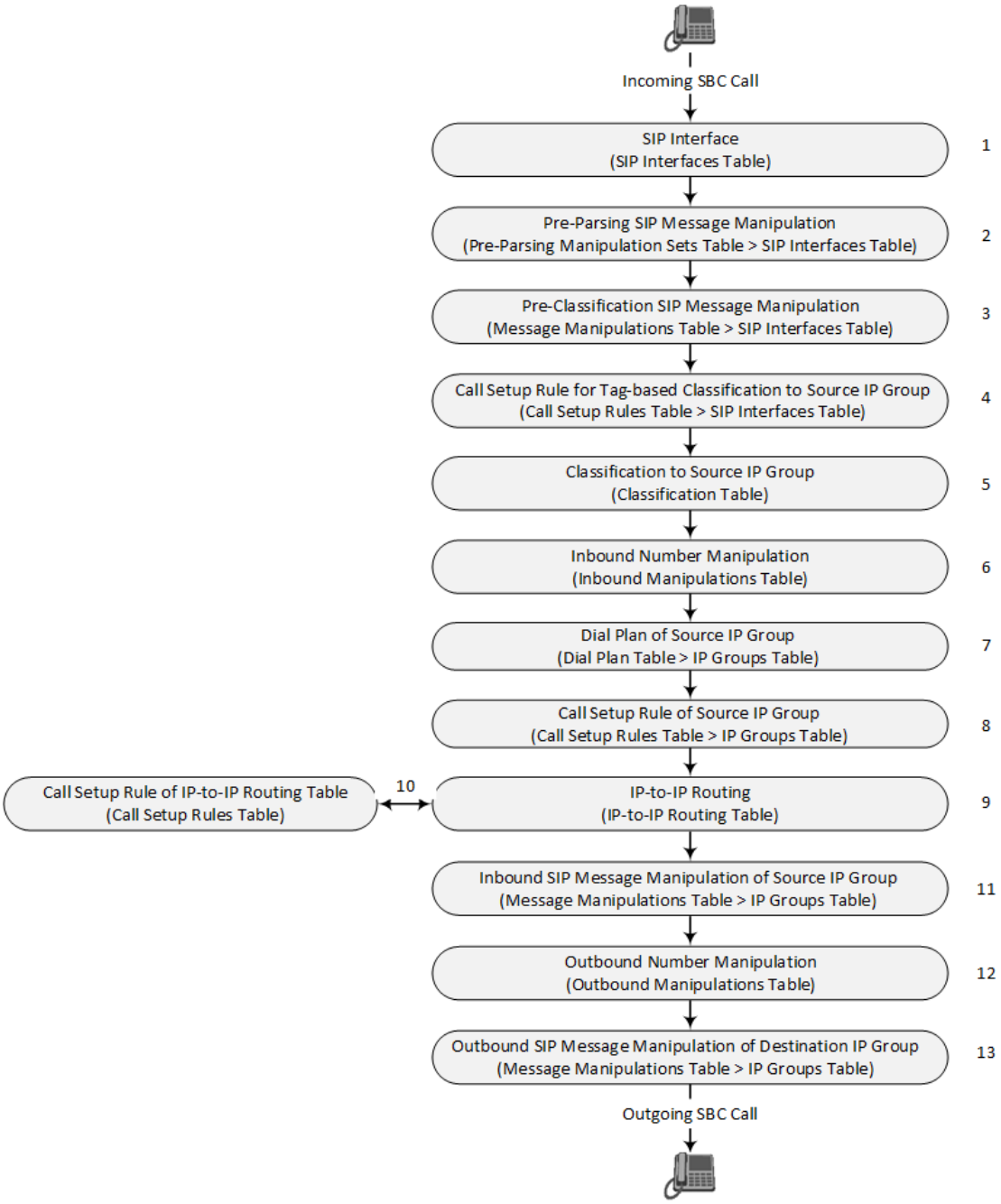
- SRDs in the SRDs table (see Configuring SRDs)
- IP Groups in the IP Groups table (see Configuring IP Groups)

If the operation mode is configured in both tables, the operation mode of the IP Group is applied. Once configured, the device uses default settings in the IP Profiles table for handling the SIP headers, as mentioned previously. However, you can change the default settings to enable partial transparency.

- The To-header tag remains the same for inbound and outbound legs of the dialog, regardless of operation mode.
- If the Operation Mode of the SRD\IP Group of one leg of the dialog is set to 'Call Stateful Proxy', the device also operates in this mode on the other leg with regards to the dialog identifiers (Call-ID header, tags, CSeq header).
- It is recommended to implement the B2BUA mode, unless one of the reasons mentioned previously is required. B2BUA supports all the device's feature-rich offerings, while Stateful Proxy may offer only limited support. The following features are not supported when in Stateful Proxy mode:
  - ✓ Alternative routing
  - ✓ Call forking
  - ✓ Terminating REFER/3xx
- If Stateful Proxy mode is enabled and any one of the unsupported features is enabled, the device disables the Stateful Proxy mode and operates in B2BUA mode.
- You can configure the device to operate in both B2BUA and Stateful Proxy modes for the same users. This is typically implemented when users need to communicate with different SIP entities (IP Groups). For example, B2BUA mode for calls destined to a SIP Trunk and Stateful Proxy mode for calls destined to an IP PBX. The configuration is done using IP Groups and SRDs.
- If Stateful Proxy mode is used only due to the debugging benefits, it is recommended to configure the device to only forward the Call-ID header unchanged

Call Processing of SIP Dialog Requests

The device processes incoming SIP dialog requests (SIP methods) such as INVITE, SUBSCRIBE, OPTIONS, REFER, INFO, UNSOLICITED NOTIFY, MESSAGE, and REGISTER. The process is summarized in the following figure and subsequently described:



The first stage of the SIP dialog-initiating process is **determining source and destination URLs**. The SIP protocol has more than one URL in a dialog-initiating request that may represent the source and destination URLs. The device obtains the source and destination URLs from certain SIP headers. Once the URLs are determined, the user and host parts of the URLs can be used as matching rule characteristics for classification, message manipulation, and call routing.

- **All SIP requests (e.g., INVITE) except REGISTER:**
  - Source URL: Obtained from the From header. If the From header contains the value 'Anonymous', the source URL is obtained from the P-Preferred-Identity header. If the P-Preferred-Identity header does not exist, the source URL is obtained from the P-Asserted-Identity header.
  - Destination URL: Obtained from the Request-URI.

- **REGISTER dialogs:**
  - Source URL: Obtained from the To header.
  - Destination URL: Obtained from the Request-URI.

You can specify the SIP header from where you want the device to obtain the source URL in the incoming dialog request. This is configured in the IP Groups table using the 'Source URI Input' parameter (see Configuring IP Groups).

The next stages of the SIP dialog-initiating process is as follows:

1. **Determining the SIP Interface:** The device checks the SIP Interface on which the SIP dialog is received. The SIP Interface defines the local SIP "listening" port and IP network interface. For more information, see Configuring SIP Interfaces.
2. **Applying Pre-parsing SIP Message Manipulation:** If configured, the device can apply SIP message manipulation to the incoming SIP message before it is parsed by the device. This type of manipulation is called Pre-Parsing Manipulation, which is configured in the Pre-Parsing Manipulation Sets table (see Configuring Pre-Parsing Manipulation Rules on page 647) and is assigned to the SIP Interface.
3. **Applying Pre-classification SIP Message Manipulation:** If configured, the device can apply SIP message manipulation to the incoming SIP message before it is classified to a source IP Group. This manipulation is configured in the SIP Message Manipulations table (see Configuring SIP Message Manipulation) and is assigned to the SIP Interface.
4. **Classifying to a Source IP Group using Tags:** If configured, the device can classify the incoming SIP message to a source IP Group, based on a source tag that is determined by running a Call Setup Rule. The Call Setup Rule is configured in the Call Setup Rules table (see Configuring Call Setup Rules on page 595) and is assigned to the SIP Interface. For more information on tag-based classification, see Configuring Classification Based on Tags on page 714.
5. **Classifying to a Source IP Group:** Classification identifies the incoming SIP dialog request as belonging to a specific IP Group (i.e., from where the SIP dialog request originated). The classification process is based on the SRD to which the dialog belongs (the SRD is determined according to the SIP Interface). For more information, see Configuring Classification Rules.
6. **Applying Inbound Manipulation:** Depending on configuration, the device can apply an Inbound Manipulation rule to the incoming dialog. This manipulates the user part of the SIP URI for source (e.g., in the SIP From header) and destination (e.g., in the Request-URI line). The manipulation rule is associated with the incoming dialog, by configuring the rule with incoming matching characteristics such as source IP Group and destination host name. The manipulation rules are also assigned a Routing Policy, which in turn, is assigned to IP-to-IP routing rules. As most deployments require only one Routing Policy, the default Routing Policy is automatically assigned to manipulation and routing rules. For more information, see Configuring IP-to-IP Inbound Manipulations.

7. **Applying a Dial Plan to Determine Tag:** If configured, the device can run a Dial Plan rule based on the source (or destination) number of the incoming SIP message to determine its' tag. The tag can later be used in the routing and manipulation stages. Dial Plan rules are configured in the Dial Plan table (see Configuring Dial Plans on page 609) and assigned to the IP Group.
8. **Applying Call Setup Rules for Various Functions:** If configured, the device can run Call Setup Rules to apply various functions to the call, for example, querying an LDAP server. The Call Setup Rule is configured in the Call Setup Rules table (see Configuring Call Setup Rules on page 595) and is assigned to the IP Group.
9. **SBC IP-to-IP Routing:** The device searches the IP-to-IP Routing table for a routing rule that matches the characteristics of the incoming call. If found, the device routes the call to the configured destination which can be, for example, an IP Group, the Request-URI if the user is registered with the device, and a specified IP address. For more information, see Configuring SBC IP-to-IP Routing Rules.
10. **Applying Call Setup Rules for Various Functions:** If configured, the device can run Call Setup Rules to apply various functions to the call. The Call Setup Rule is configured in the Call Setup Rules table (see Configuring Call Setup Rules on page 595) and is assigned to the IP-to-IP Routing table.
11. **Applying Inbound SIP Message Manipulation:** Depending on configuration, the device can apply a SIP message manipulation rule (assigned to the IP Group) on the incoming dialog. For more information, see Stage 3.
12. **Applying Outbound Manipulation:** Depending on configuration, the device can apply an Outbound Manipulation rule to the outbound dialog. This manipulates the user part of the Request-URI for source (e.g., in the SIP From header) or destination (e.g., in the SIP To header) or calling name in the outbound SIP dialog. The manipulation rule is associated with the dialog, by configuring the rule with incoming matching characteristics such as source IP Group and destination host name. The manipulation rules are also assigned a Routing Policy, which in turn, is assigned to IP-to-IP routing rules. As most deployments require only one Routing Policy, the default Routing Policy is automatically assigned to manipulation rules and routing rules. For more information, see Configuring IP-to-IP Outbound Manipulations.
13. **Applying Outbound SIP Message Manipulation:** Depending on configuration, the device can apply a SIP message manipulation rule (assigned to the IP Group) on the outbound dialog. For more information, see Stage 3.
14. The call is sent to the configured destination.

User Registration

The device provides a registration database for registering users. Only users belonging to a User-type IP Group can register with the device. User-type IP Groups represent a group of SIP user agents that share the following characteristics:

- Perform registrations and share the same serving proxy\registrar

- Same SIP and media behavior
- Same IP Profile
- Same SIP handling configuration
- Same Call Admission Control (CAC)

Typically, the device is configured as the user's outbound proxy, routing requests (using the IP-to-IP Routing table) from the user's User-type IP Group to the serving proxy, and vice versa. Survivability can be achieved using the alternative routing feature.

The device forwards registration requests (REGISTER messages) from a Server-type IP Group, but does not save the registration binding in its' registration database.

Initial Registration Request Processing

A summary of the device's handling of registration requests (REGISTER messages) is as follows:

- The URL in the SIP To header of the REGISTER message constitutes the primary Address of Record (AOR) for registration (according to SIP standards). If the To header's URL includes the "user=phone" parameter, then only the user part of the URL constitutes the AOR. If the To header's URL does not include the "user=phone" parameter, then both the user part and host part of the URL constitutes the AOR.
- The device can save other AORs in its registration database as well. When the device searches for a user in its' registration database, any of the user's AORs can result in a match.
- The device's Classification process for initial REGISTER messages is slightly different than for other SIP messages. Unlike other requests, initial REGISTER requests can't be classified according to the registration database.
- If registration succeeds (replied with 200 OK by the destination server), the device adds a record to its' registration database, which identifies the specific contact of the specific user (AOR). The device uses this record to route subsequent SIP requests to the specific user (in normal or survivability modes).
- The records in the device's registration database include the Contact header. The device adds every REGISTER request to the registration database before manipulation, allowing correct user identification in the Classification process for the next received request.
- You can configure Call Admission Control (CAC) rules for incoming and outgoing REGISTER messages. For example, you can limit REGISTER requests from a specific IP Group or SRD. Note that this applies only to concurrent REGISTER dialogs and not concurrent registrations in the device's registration database.

The device provides a dynamic registration database that it updates according to registration requests traversing it. Each database entry for a user represents a binding between an AOR (obtained from the SIP To header), optional additional AORs, and one or more contacts (obtained from the SIP Contact headers). Database bindings are added upon successful

registration responses from the proxy server (SIP 200 OK). The device removes database bindings in the following cases:

- Successful de-registration responses (REGISTER with Expires header that equals zero).
- Registration failure responses.
- Timeout of the Expires header value (in scenarios where the UA did not send a refresh registration request).

- The same contact cannot belong to more than one AOR.
- Contacts with identical URIs and different ports and transport types are not supported (same key is created).
- Multiple contacts in a single REGISTER message is not supported.
- One database is shared between all User-type IP Groups.

Classification and Routing of Registered Users

The device can classify incoming SIP dialog requests (e.g., INVITE) from registered users to an IP Group, by searching for the sender's details in the registration database. The device uses the AOR from the From header and the URL in the Contact header of the request to locate a matching registration binding. The found registration binding contains information regarding the registered user, including the IP Group to which it belongs. (Upon initial registration, the Classification table is used to classify the user to a User-type IP Group and this information is then added with the user in the registration database.)

The destination of a dialog request can be a registered user and the device thus uses its registration database to route the call. This can be achieved by various ways such as configuring a rule in the IP-to-IP Routing table where the destination is a User-type IP Group or any matching user registered in the database ('Destination Type' is configured to **All Users**). The device searches the registration database for a user that matches the incoming Request-URI (listed in chronological order):

- Unique Contact generated by the device and sent in the initial registration request to the serving proxy.
- AOR. The AOR is originally obtained from the incoming REGISTER request and must either match both user part and host part (user@host) of the Request-URI, or only user part.
- Contact. The Contact is originally obtained from the incoming REGISTER request.

If registrations are destined to the database (using the above rules), the device does not attempt to find a database match, but instead replies with a SIP 200 OK (used for Survivability). Once a match is found, the request is routed either to the contact received in the initial registration or (if the device identifies that the user agent is behind a NAT) to the source IP address of the initial registration.

You can configure (using the [SBCDBRoutingSearchMode] parameter) for which part of the destination Request-URI in the INVITE message the device must search in the registration database:

- Only by entire Request-URI (user@host), for example, "4709@joe.company.com".
- By entire Request-URI, but if not found, by the user part of the Request-URI, for example, "4709".

When an incoming INVITE is received for routing to a user and the user is located in the registration database, the device sends the call to the user's corresponding contact address specified in the database.

> **Note:** If the Request-URI contains the "tel:" URI or "user=phone" parameter, the device searches only for the user part.

You can also configure (using the [SBCURIComparisonExcludedParams] parameter) which URI parameters are excluded when the device compares the URIs of two incoming dialog-initiating SIP requests (e.g., INVITEs) to determine if they were sent from a user that is registered in the device's registration database. For example, you can configure the parameter to exclude ports from the comparison. For more information, see the description of the [SBCURIComparisonExcludedParams] parameter.

General Registration Request Processing

The device's general handling of registration requests (REGISTER messages) for unregistered users is as follows:

- The device routes REGISTER requests according to the IP-to-IP Routing table. If the destination is a User-type IP Group, the device does not forward the registration; instead, it accepts (replies with a SIP 200 OK response) or rejects (replies with a SIP 4xx) the request according to the user's IP Group configuration.
- Alternative routing can be configured for REGISTER requests, in the IP-to-IP Routing table.
- By default, the Expires header has the same value in incoming and outgoing REGISTER messages. However, you can modify the Expires value using the following parameters: SBCUserRegistrationTime, SBCProxyRegistrationTime, SBCRandomizeExpires, and SBCSurvivabilityRegistrationTime. You can also modify the Expires value of REGISTER requests received from users located behind NAT, using the IP Profile parameters IpProfile_SBCUserBehindUdpNATRegistrationTime and IpProfile_SBCUserBehindTcpNATRegistrationTime.
- By default, the Contact header in outgoing REGISTER message is different than the Contact header in the incoming REGISTER. The user part of the Contact is populated with a unique contact generated by the device and associated with the specific registration. The IP address in the host part is changed to the address of the device. Alternatively, the original user can be retained in the Contact header and used in the outgoing REGISTER request (using the SBCKeepContactUserinRegister parameter).

Registration Refreshes

Registration refreshes are incoming REGISTER requests from users that are registered in the device's registration database. The device sends these refreshes to the serving proxy only if the serving proxy's Expires time is about to expire; otherwise, the device responds with a 200 OK to the user without routing the REGISTER. Each such refresh also refreshes the internal timer set on the device for this specific registration.

The device automatically notifies SIP proxy / registrar servers of users that are registered in its registration database and whose registration timeout has expired. When a user's registration timer expires, the device removes the user's record from the database and sends an un-register notification (REGISTER message with the Expires header set to 0) to the proxy/registrar. This occurs only if a REGISTER message is sent to an IP Group destination type (in the IP-to-IP Routing table).

You can also apply a graceful period to unregistered requests, using the 'User Registration Grace Time' parameter ([SBCUserRegistrationGraceTime]):

- You can configure the device to add extra time (grace period) to the expiration timer of registered users in the database. If you configure this grace period, the device keeps the user in the database (and does not send an unregister to the registrar server), allowing the user to send a "late" re-registration to the device. The device removes the user from the database only when this additional time expires.
- The graceful period is also used before removing a user from the registration database when the device receives a successful unregister response (200 OK) from the registrar/proxy server. This is useful in scenarios, for example, in which users (SIP user agents) such as IP Phones erroneously send unregister requests. Instead of immediately removing the user from the registration database upon receipt of a successful unregister response, the device waits until it receives a successful unregister response from the registrar server, waits the user-defined graceful time and if no register refresh request is received from the user agent, removes the contact (or AOR) from the database.

The device keeps registered users in its' registration database even if connectivity with the proxy is lost (i.e., proxy does not respond to users' registration refresh requests). The device removes users from the database only when their registration expiry time is reached (with the additional grace period, if configured).

Registration Restriction Control

The device provides flexibility in controlling user registrations:

- **Limiting Number of Registrations:** You can limit the number of users that can register with the device per IP Group, SIP Interface, and/or SRD, in the IP Group, SIP Interface and SRDs tables respectively. By default, no limitation exists.
- **Blocking Incoming Calls from Unregistered Users:** You can block incoming calls (INVITE requests) from unregistered users belonging to User-type IP Groups. By default, calls from unregistered users are not blocked. This is configured per SIP Interface or SRD. When the

call is rejected, the device sends a SIP 500 (Server Internal Error) response to the remote end.

Deleting Registered Users

You can remove registered users from the device's registration database through CLI:

- To delete a specific registered user:

```
# clear voip register db sbc user <AOR of user - user part or user@host>
```

For example:

```
# clear voip register db sbc user John@10.33.2.22
# clear voip register db sbc user John
```

- To delete all registered users belonging to a specific IP Group:

```
# clear voip register db sbc ip-group <ID or name>
```

Media Handling

Media behavior includes anything related to the establishment, management and termination of media sessions within the SIP protocol. Media sessions are created using the SIP offer-answer mechanism. If successful, the result is a bi-directional media (RTP) flow (e.g. audio, fax, modem, DTMF). Each offer-answer may create multiple media sessions of different types (e.g. audio and fax). In a SIP dialog, multiple offer-answer transactions may occur and each may change the media session characteristics (e.g. IP address, port, coders, media types, and RTP mode). The media capabilities exchanged in an offer-answer transaction include the following:

- Media types (e.g., audio, secure audio, video, fax, and text)
- IP addresses and ports of the media flow
- Media flow mode (send receive, receive only, send only, inactive)
- Media coders (coders and their characteristics used in each media flow)
- Other (standard or proprietary) media and session characteristics

Typically, the device does not change the negotiated media capabilities (mainly performed by the remote user agents). However, it does examine and may take an active role in the SDP offer-answer mechanism. This is done mainly to anchor the media to the device (default) and also to change the negotiated media type, if configured. Some of the media handling features, which are described later in this section, include the following:

- Media anchoring (default)
- Direct media

- Audio coders restrictions
- Audio coders transcoding
- RTP-SRTP transcoding
- DTMF translations
- Fax translations and detection
- Early media and ringback tone handling
- Call hold translations and held tone generation
- NAT traversal
- RTP broken connections
- Media firewall
  - RTP pin holes - only RTP packets related to a successful offer-answer negotiation traverse the device: When the device initializes, there are no RTP pin holes opened. This means that each RTP\RTCP packets destined to the device are discarded. Once an offer-answer transaction ends successfully, an RTP pin hole is opened and RTP\RTCP flows between the two remote user agents. Once a pin hole is opened, the payload type and RTP header version is validated for each packet. RTP pin holes close if one of the associated SIP dialogs is closed (may also be due to broken connection).
  - Late rogue detection - once a dialog is disconnected, the related pin holes also disconnect.
  - Deep Packet inspection of the RTP that flows through the opened pin holes.

Media Anchoring

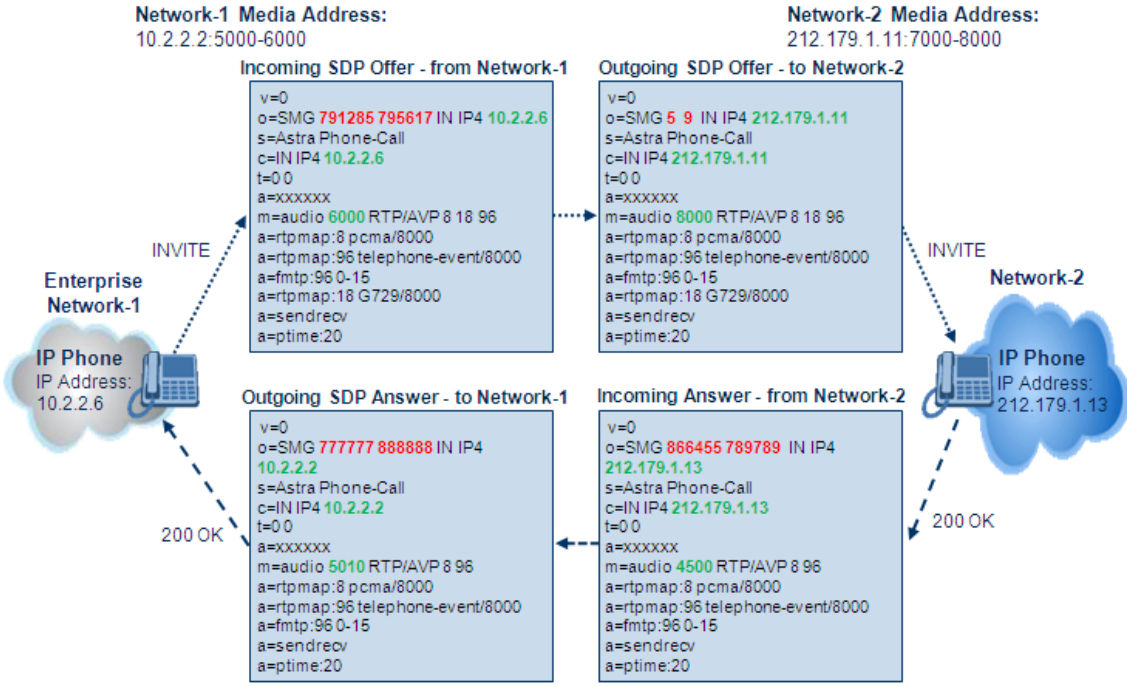
By default, the device anchors the media (RTP) traffic. In other words, the media between SIP endpoints traverses the device. You can change this default mode by enabling direct media between SIP endpoints. Media anchoring may be required, for example, to resolve NAT problems, enforce media security policies, perform media transcoding, and media monitoring.

To enforce RTP traffic to flow through the device, the device modifies all IP address fields in the SDP:

- Origin: IP address, session and version id
- Session connection attribute ('c=' field)
- Media connection attribute ('c=' field)
- Media port number
- RTCP media attribute IP address and port

The device uses different local ports (e.g., for RTP, RTCP and fax) for each leg (inbound and outbound). The local ports are allocated from the Media Realm associated with each leg. The Media Realm assigned to the leg's IP Group (in the IP Groups table) is used. If not assigned to

the IP Group, the Media Realm assigned to the leg's SIP Interface (in the SIP Interfaces table) is used. The following figure provides an example of SDP handling for a call between a LAN IP Phone 10.2.2.6 and a remote IP Phone 212.179.1.13 on the WAN.



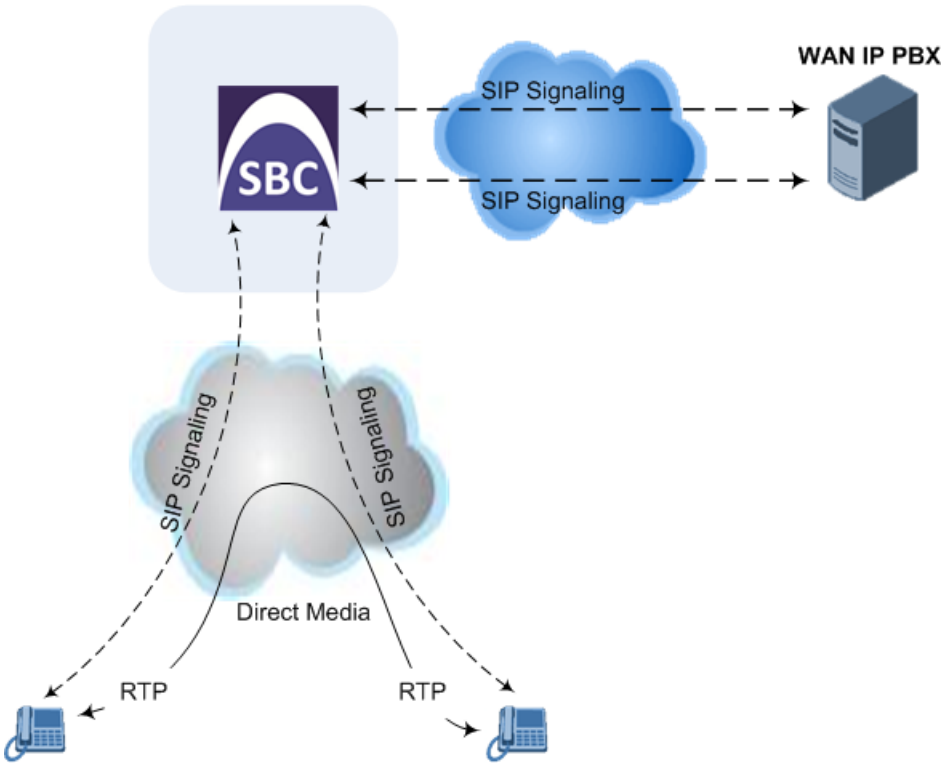
Direct Media Calls

You can configure the device to allow the media (RTP/SRTP) session to flow directly between the SIP endpoints without traversing the device. This is referred to as No Media Anchoring (also known as Anti-Tromboning or Direct Media). SIP signaling continues to traverse the device with minimal intermediation and involvement to enable certain SBC capabilities such as routing. By default, the device employs media anchoring, whereby the media session traverses the device, as described in Media Anchoring.

Direct media offers the following benefits:

- Saves network bandwidth
- Reduces the device's CPU usage (as there is no media handling)
- Avoids interference in SDP negotiation and header manipulation on RTP/SRTP

Direct media is typically implemented for calls between users located in the same LAN or domain, and where NAT traversal is not required and other media handling features such as media transcoding is not required. The following figure provides an example of direct media between LAN IP phones, while SIP signaling continues to traverse the device between LAN IP phones and the hosted WAN IP-PBX.



➢ **To enable direct media:**

- **For all calls:** Use the global parameter [SBCDirectMedia], which **overrides** all other direct media configuration.
- **For specific calls:**
  - SIP Interface: You can enable direct media per SIP Interface (in the SIP Interfaces table), whereby calls (source and destination) associated with **this same** SIP Interface are handled as direct media calls. The SIP Interface can also enable direct media for users located behind the same NAT. For more information, see Configuring SIP Interfaces.
  - Direct Media Tag: You can enable direct media between users that are configured with the same Direct Media tag value. The tag is configured using the IP Profiles table's IPProfile_SBCDirectMediaTag parameter (see Configuring IP Profiles).

The device employs direct media between endpoints under the following configuration conditions (listed in chronological order):

1. Direct media is enabled by the global parameter [SBCDirectMedia].
2. IP Groups of the endpoints are associated with IP Profiles whose 'Direct Media Tag' parameter has the same value (non-empty value).
3. IP Groups of the endpoints have the 'SBC Operation Mode' parameter set to **Microsoft Server** (direct media is required in the Skype for Business environment). For more information, see Configuring IP Groups.
4. IP Groups of the endpoints use the same SIP Interface and the SIP Interface's 'SBC Direct Media' parameter is set to **Enable** (SIPInterface_SBCDirectMedia = 1).

5. IP Groups of the endpoints use the same SIP Interface and the SIP Interface's 'SBC Direct Media' parameter is set to Enable When Single NAT (SIPInterface_SBCDirectMedia = 2), and the endpoints are located behind the same NAT.

- **Direct Media configured for all calls (i.e., using the [SBCDirectMedia] parameter):** The device doesn't open voice channels and doesn't allocate media ports for these calls, because the media always bypasses the device.
- **Direct Media configured for specific calls (i.e., using the IP Profile's 'Direct Media Tag' parameter or SIP Interface's 'Direct Media' parameter):** The device always allocates ports for these calls, because these ports may be required at some stage during the call if it changes to a **non-direct media** call for mid-call services such as early media, call forwarding, call transfer, or playing on-hold tones. Therefore, make sure that you have allocated sufficient media ports (Media Realm) for these calls.
- The following features are not supported for Direct Media calls:
  - ✓ Manipulation of SDP data (offer-answer transaction) such as ports, IP address, coders
  - ✓ Forced transcoding
  - ✓ Extension Coders
  - ✓ Extension of RFC 2833 / out-of-band DTMF / in-band DTMF
  - ✓ Extension of SRTP/RTP
  - ✓ All restriction features (Allowed Coders, restrict SRTP/RTP, and restrict RFC 2833)
  - ✓ All media-related parameters in the IP Profiles table are not applicable to Direct Media calls
- The device doesn't fully support call transfer (SIP REFER) terminations for direct media calls. One of the SIP User Agents (UA) in the call must support re-INVITE messages without SDP for the device to synchronize the media.
- For two users belonging to the same SIP Interface that is enabled for direct media and one of the users is defined as a foreign user (example, "follow me service") located in the WAN while the other is located in the LAN: calls between these two users cannot be established until direct media is disabled for the SIP Interface. The reason for this is that the device does not interfere in the SIP signaling. In other words, parameters such as IP addresses are not manipulated for calls between LAN and WAN (although required).
- If you have configured a SIP Recording rule (see SIP-based Media Recording on page 247) for calls that have also been configured for direct media, using a SIP Interface ('Direct Media' parameter) or an IP Profile ('Direct Media Tag' parameter), the device automatically disables direct media for these calls (during their SIP signaling setup). This ensures that the media passes through the device so that it can be recorded and sent to the SRS. However, if you enable direct media using the [SBCDirectMedia] parameter (i.e., for all calls), direct media is always enforced and calls will not be recorded.

Restricting Audio Coders

You can configure a list of permitted (allowed) voice coders that can be used for a specific SIP entity (leg). In other words, you can enforce the use of specific coders. If the SDP offer in the incoming SIP message does not contain any coder that is configured as an allowed coder, the

device rejects the calls (unless transcoding is implemented whereby Extension coders are added to the SDP, as described in Coder Transcoding). If the SDP offer contains some coders that are configured as allowed coders, the device manipulates the SDP offer by removing the coders that are not configured as allowed coders, before routing the SIP message to its destination. The device also re-orders (prioritizes) the coder list in the SDP according to the listed order of configured allowed coders.

For example, assume the following:

- The SDP offer in the incoming SIP message contains the G.729, G.711, and G.723 coders.
- The allowed coders configured for the SIP entity include G.711 and G.729.

The device removes the G.723 coder from the SDP offer, re-orders the coder list so that G.711 is listed first, and sends the SIP message containing only the G.711 and G.729 coders in the SDP.

The allowed coders are configured in the Allowed Audio Coders Groups table. For more information, see Configuring Allowed Audio Coder Groups.

> **Note:** If you assign the SIP entity an Allowed Audio Coders Group for coder restriction and a Coders Group for extension coders (i.e., voice transcoding), the allowed coders take precedence over the extension coders. In other words, if an extension coder is not listed as an allowed coder, the device does not add the extension coder to the SDP offer.

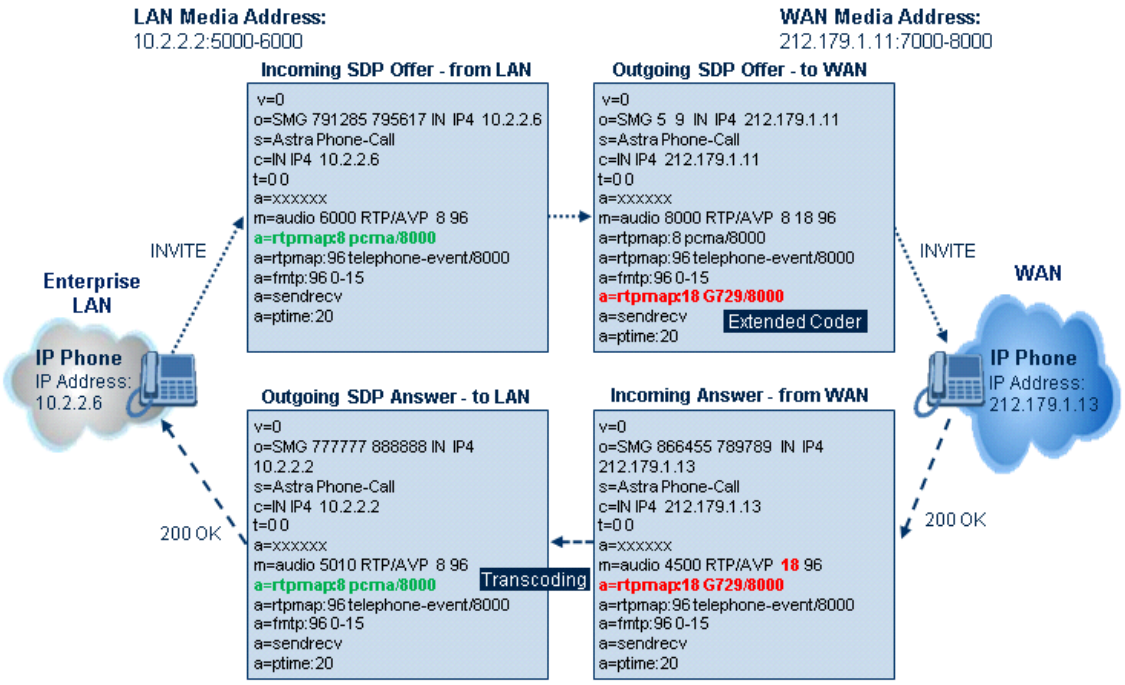
Coder Transcoding

By default, the device forwards media packets transparently (i.e., no media negotiation) between the SIP endpoints. However, when there are no common coders between two SIP entities that need to establish voice communication (i.e., the SDP answer from one SIP entity doesn't include any coder included in the SDP offer previously sent by the other), you can configure the device to perform audio coder transcoding between the inbound and outbound legs in order to enable media flow between them.

Transcoding may also be performed in scenarios where the same coder has been chosen between the legs, but where coder transrating is required. For example, the coders may use different coder settings such as rate and packetization time (G.729 at 20 ms to G.729 at 30 ms).

The coders that the device adds to the SDP offer on the outbound leg is referred to as *extension coders*. The extension coders are configured using Coder Groups (see Configuring Coder Groups), which you need to then assign to the IP Profile associated with the SIP entity.

The figure below illustrates transcoding between two SIP entities (IP Groups) where one uses G.711 (LAN IP phone) and the other G.729 (WAN IP phone). The initial SDP offer received on the inbound leg from the LAN IP phone includes coder G.711 as the supported coder. In the outgoing SDP offer on the outbound leg to the WAN IP phone, the device adds extension coder G.729 to the SDP, which is supported by the WAN IP phone. The subsequent incoming SDP answer from the WAN IP phone includes the G.729 coder as the chosen coder. Since this coder was not included in the original incoming SDP offer from the LAN IP phone, the device performs G.729-G.711 transcoding between the inbound and outbound legs.



- If you assign a SIP entity an Allowed Audio Coders Group for coder restriction (allowed coders) and a Coders Group for extension coders, the allowed coders take precedence over the extension coders. In other words, if an extension coder is not listed as an allowed coder, the device does not add the extension coder to the SDP offer.
- If none of the coders in the incoming SDP offer on the inbound leg appear in the associated Allowed Audio Coders Group for coder restriction, the device rejects the call (sends a SIP 488 to the SIP entity that initiated the SDP offer).
- If none of the coders (including extension coders) in the outgoing SDP offer on the outbound leg appear in the associated Allowed Audio Coders Group for coder restriction, the device rejects the call (sends a SIP 488 to the SIP entity that initiated the SDP offer).
- For coder transcoding, the following prerequisites must be met (otherwise, the extension coders are not added to the SDP offer):
  - ✓ The device must support at least one of the coders listed in the incoming SDP offer.
  - ✓ The device must have available DSPs for both legs (inbound and outbound).
  - ✓ The incoming SDP offer must have at least one media line that is audio ('m=audio').
- The device adds the extension coders below the coder list received in the original SDP offer. This increases the chance of media flow without requiring transcoding.
- The device does not add extension coders that also appear in the original SDP offer.
- You can view the number of currently active transcoding sessions, using the CLI command `show voip calls statistics sbc media`.

As an example for using allowed and extension coders, assume the following:

- **Inbound leg:**

- Incoming SDP offer includes the G.729, G.711, and G.723 coders.

```
m=audio 6050 RTP/AVP 18 0 8 4 96
a=rtpmap:18 G729/8000
a=rtpmap:0 PCMU/8000
a=rtpmap:8 PCMA/8000
a=rtpmap:4 G723/8000
a=fmtp:4 annexa=no
a=rtpmap:96 telephone-event/8000
a=fmtp:96 0-15
a=ptime:20
a=sendrecv
```

  The SDP "m=audio 6010 RTP/AVP 18 0 8 4 96" line shows the coder priority, where "18" (G.729) is highest and "4" (G.723) is lowest.

- Allowed Audio Coders Group for coder restriction includes the G.711 and G.729 coders (listed in order of appearance).

Outbound leg:

- Allowed Audio Coders Group for coder restriction includes the G.723, G.726, and G.729 coders (listed in order of appearance).
- Allowed Audio Coders Group for coder extension (transcoding) includes the G.726 coder.

1. On the inbound leg for the incoming SDP offer: The device allows and keeps the coders in the SDP that also appear in the Allowed Audio Coders Group for coder restriction (i.e., G.711 and G.729). It changes the order of listed coders in the SDP so that G.711 is listed first. The device removes the coders (i.e., G.723) from the SDP that do not appear in the Allowed Audio Coders Group for coder restriction.

```
m=audio 6050 RTP/AVP 0 8 18 96
a=rtpmap:0 PCMU/8000
a=rtpmap:8 PCMA/8000
a=rtpmap:18 G729/8000
a=fmtp:4 annexa=no
a=rtpmap:96 telephone-event/8000
a=fmtp:96 0-15
a=ptime:20
a=sendrecv
```

2. On the outbound leg for the outgoing SDP offer: The SDP offer now includes only the G.711 and G.729 coders due to the coder restriction process on the incoming SDP offer (see Step 1).

a. The device adds the extension coder to the SDP offer and therefore, the SDP offer now includes the G.711, G.729 and G.726 coders.

```
m=audio 6050 RTP/AVP 0 8 18 96 96
a=rtpmap:0 PCMU/8000
a=rtpmap:8 PCMA/8000
a=rtpmap:18 G729/8000
a=rtpmap:96 G726-32/8000
a=fmtp:4 annexa=no
a=rtpmap:96 telephone-event/8000
a=fmtp:96 0-15
a=ptime:20
a=sendrecv
```

b. The device applies coder restriction to the SDP offer. As the Allowed Audio Coders Group for coder restriction includes the G.723, G.726, and G.729 coders, the device allows and keeps the G.729 and G.726, but removes the G.711 coder as it does not appear in the Allowed Audio Coders Group for coder restriction.

```
m=audio 6050 RTP/AVP 18 96 96
a=rtpmap:18 G729/8000
a=rtpmap:96 G726-32/8000
a=fmtp:4 annexa=no
a=rtpmap:96 telephone-event/8000
a=fmtp:96 0-15
a=ptime:20
a=sendrecv
```

3. The device includes only the G.729 and G.726 coders in the SDP offer that it sends from the outgoing leg to the outbound SIP entity. The G.729 is listed first as the Allowed Audio Coders Group for coder restriction takes precedence over the extension coder.

➢ **To configure coder transcoding:**

1. In the Coder Groups table, configure a Coders Group for extension coders. For more information, see Configuring Coder Groups.
2. In the IP Profiles table, configure the IP Profile associated with the SIP entity:
   a. Assign the Coders Group to the IP Profile, using the 'Extension Coders Group' parameter (SBCExtensionCodersGroupName).
   b. Enable extension coders by configuring the 'Allowed Coders Mode' parameter to **Restriction** or **Restriction and Preference**.

- The device's License Key (see License Key on page 871) specifies transcoding capabilities:
  - ✓ 'DSP Channels' - maximum number of DSP resources.
  - ✓ 'Transcoding Sessions' - maximum number of transcoding sessions.
- Each transcoding session uses two DSP resources.
- You can configure the transcoding mode globally, using the [TranscodingMode] parameter, or for specific calls, using the IP Profiles table parameter [IpProfile_TranscodingMode].

Transcoding Mode

By default, the device performs transcoding only when necessary. This refers to all types of transcoding (interworking) that require the use of the device's DSP resources, for example, voice coder transcoding, DTMF negotiations, and fax negotiations. Transcoding is required, for example, when two SIP entities use different coders. In such a scenario, the device can be configured to use a different coder for each leg (inbound and outbound), using IP Profiles. If the SIP entities use the same coder, the device does not perform transcoding.

Alternatively, you can configure the device to always perform transcoding, regardless of whether it is required or not. This is referred to as *forced* transcoding. For example, if the SIP entities use the same coder, the device performs transcoding of the same coder (e.g., G.711 and G.711) between the two legs.

To configure the transcoding mode, use the global parameter [TranscodingMode] or the IP Profile parameter [IpProfile_TranscodingMode] parameter.

If the transcoding mode is configured to **Force Transcoding** (i.e., always perform transcoding) for an IP Profile associated with a specific SIP entity, the device also applies forced transcoding for the SIP entity communicating with this SIP entity, regardless of its IP Profile settings.

Prioritizing Coder List in SDP Offer

In addition to restricting the use of coders using Allowed Audio Coders Groups (see Configuring Allowed Audio Coder Groups), you can also prioritize the coders listed in the SDP offer. This feature is referred to as *Coder Preference* and applies to both SBC legs:

- **Incoming SDP offer:** The device arranges the coder list in the incoming SDP offer according to the order of appearance of the Allowed Audio Coders Group that is associated with the incoming dialog. The coders listed higher up in the group take preference over ones listed lower down. To configure this, configure the 'Allowed Coders Mode' parameter (IpProfile_SBCAllowedCodersMode) in the associated IP Profile to **Preference** or **Restriction and Preference**. If you configure the parameter to to **Preference**, the coders in the SDP offer that also appear in the Allowed Audio Coders Group are listed first in the SDP offer, and the coders in the SDP offer that do not appear in the Allowed Audio Coders Group are listed

after the Allowed coders in the SDP offer. Therefore, this setting does not restrict coder use to Allowed coders, but uses (prefers) the Allowed coders whenever possible.

- **Outgoing SDP offer:** If only Allowed coders are used, the device arranges the coders in the SDP offer as described above. However, if Extension coders are also used, the coder list is arranged according to the SBCPreferencesMode parameter. Depending on the parameter's settings, the Extension coders are added after the Allowed coders according to their order in the Allowed Audio Coders Group, or the Allowed and Extension coders are arranged according to their position in the Allowed Audio Coders Group.

Allocating DSPs on SDP Offer or Answer

By default, the device allocates DSP resources for a call at the SDP Offer stage. If DSP resources are available at this stage, the device reserves DSPs for the call just in case call setup succeeds with the SDP Answer and DSPs are required (e.g., for transcoding). If there are no free DSP resources at the SDP Offer stage, no DSP resources are allocated for the call, at any stage of the SDP Offer-Answer exchange, and if DSPs are required (based on the SDP Answer), the device rejects the call.

However, this default behavior may cause call failure for a call requiring DSPs even when the device has sufficient DSP resources. For example, assume the device is licensed for 10 concurrent transcoding calls and is currently handling the establishment of 10 calls where only half require transcoding (DSPs). For all these calls, the device allocates DSPs during the SDP Offer stage (even if some of these calls may not require DSPs, based on the SDP Answer). If during this time the device starts processing an 11th call that requires transcoding (DSPs), since it has already allocated all of its DSP resources, it doesn't allocate any DSPs to this call and as a result, the device rejects the call.

To avoid such scenarios, you can configure the device to allocate DSPs only at the SDP Answer stage (SIP 200 OK or 180), when it can determine if DSPs are required or not for the call. If DSPs are required and DSP resources are available, the device allocates DSPs. If DSPs are required but there are no available DSPs, the device rejects the call.

➢ **To disable reserving DSPs on SDP Offer:**

1. Open the Media Settings page (**Setup** menu > **Signaling & Media** tab > **Media** folder > **Media Settings**).
2. From the 'Reserve DSP on SDP Offer' drop-down list, select **Disable**.

3. Click **Apply**.

SRTP-RTP and SRTP-SRTP Transcoding

The device supports transcoding between SRTP and RTP. The device can also enforce specific SBC legs to use SRTP and/or RTP. The device's handling of SRTP/RTP is configured using the IP

Profile parameter [SBCMediaSecurityBehaviour], which provides the following options:

- SBC passes the media as is, regardless of whether it's RTP or SRTP (default).
- SBC legs negotiate only SRTP media lines (m=); RTP media lines are removed from the incoming SDP offer-answer.
- SBC legs negotiate only RTP media lines; SRTP media lines are removed from the incoming offer-answer.
- Each SDP offer-answer is extended (if not already) to two media lines for RTP and SRTP.

If after SDP offer-answer negotiation, an SBC leg uses RTP while the other uses SRTP, the device performs RTP- SRTP transcoding. To translate between RTP and SRTP, the following prerequisites must be met:

- At least one supported SDP "crypto" attribute.
- SRTP must be enabled - [EnableMediaSecurity] parameter configured to [1].

Transcoding where both legs are configured for SRTP is typically required to trans-encrypt and trans-decrypt. This is relevant when the MKI and Symmetric MKI parameters are enabled. In other words, both sides need to both encrypt and decrypt the outgoing and incoming SRTP packets, respectively.

DSP resources are not required for RTP-SRTP transcoding.

Multiple RTP Media Streams per Call Session

The device's SBC application supports multiple RTP media streams per SBC call session. It supports the negotiation of up to five media streams ('m=' line) in the SDP offer/answer model per session. The media can include a combination of any of the following types:

- Audio, indicated in the SDP as 'm=audio'
- Video, indicated in the SDP as 'm=video'
- Text, indicated in the SDP as 'm=text'
- Fax, indicated in the SDP as 'm=image'
- Binary Floor Control Protocol (BFCP), indicated in the SDP as 'm=application <port> UDP/BFCP'

Therefore, the device supports transcoding of various attributes in the SDP offer-answer (e.g., codec, port, and packetization time) per media type. If the device is unable to perform transcoding (e.g., does not support the coder), it relays the SBC dialog transparently.

The device transparently forwards Binary Floor Control Protocol (BFCP) signaling over UDP between IP entities (RFC 4582). BFCP is a signaling protocol used by some third- party conferencing servers to share content (such as video conferencing, presentations or documents) between conference participants (SIP clients supporting BFCP). The SDP offer/answer exchange model is used to establish (negotiate) BFCP streams between clients.

The BFCP stream is identified in the SDP as 'm=application <port> UDP/BFCP' and a dedicated UDP port is used for the BFCP streams.

Interworking Miscellaneous Media Handling

This section describes various interworking features relating to media handling.

Interworking DTMF Methods

The device supports interworking between various DTMF methods such as RFC 2833, In-Band DTMF's, and SIP INFO (Cisco\Nortel\Korea). By default, the device allows the remote user agents to negotiate (in case of RFC 2833) and passes DTMF without intervention. However, if two user agents (UA) support different DTMF methods, the device can interwork these different DTMF methods at each leg.

This DTMF interworking feature is enabled using IP Profiles (*ini* file parameter IPProfile):

- SBCRFC2833Behavior - affects the RFC 2833 SDP offer-answer negotiation:
  - [0]: (default) the device does not intervene in the RFC 2833 negotiation.
  - [1]: each outgoing offer-answer includes RFC 2833 in the offered SDP (the device adds RFC 2833 only if the incoming offer does not include RFC 2833).
  - [2]: the device removes RFC 2833 from the incoming offer.
- SBCAlternativeDTMFMethod – the device's first priority for DTMF method at each leg is RFC 2833. Therefore, if a specific leg negotiates RFC 2833 successfully, then the chosen DTMF method for this leg is RFC 2833. For legs where RFC 2833 is not negotiated successfully, the device uses the parameter to determine the DTMF method for the leg.

The chosen DTMF method determines (for each leg) which DTMF method is used for sending DTMF's. If the device interworks between different DTMF methods and one of the methods is In-band\RFC 2833, detection and generation of DTMF methods requires DSP resources.

Interworking RTP Redundancy

The device supports interworking of RTP redundancy (according to RFC 2198) between SIP entities. Employing IP Profiles, you can configure RTP redundancy handling per SIP entity:

- Generate RFC 2198 redundant packets (IpProfile_RTPRedundancyDepth parameter).
- Determine RTP redundancy support in the RTP redundancy negotiation in SDP offer/answer (IpProfile_SBCRTPRedundancyBehavior parameter). If not supported, the device discards RTP redundancy packets (if present) received from or sent to the SIP entity.

For more information, see the above parameters in Configuring IP Profiles.

Interworking RTP-RTCP Multiplexing

The device supports interworking of RTP-RTCP multiplexing onto a single, local UDP port (according to RFC 5761) between SIP entities. Employing IP Profiles, you can configure RTP

multiplexing per SIP entity, using the IPProfile_SBCRTCPMux parameter (see Configuring IP Profiles).

Interworking RTCP Attribute in SDP

The device supports interworking the RTCP attribute 'a=rtcp' in the SDP between SIP entities. Employing IP Profiles, you can configure RTCP attribute handling (add, remove or transparent) per SIP entity, using the IpProfile_SBCSDPHandleRTCPAttribute parameter (see Configuring IP Profiles).

Interworking Crypto Lifetime Field

The device supports interworking the lifetime field in the 'a=crypto' attribute of the SDP, between SIP entities. Employing IP Profiles, you can configure the lifetime field handling (remove or retain) per SIP entity, using the IpProfile_ SBCRemoveCryptoLifetimeInSDP parameter (see Configuring IP Profiles).

Interworking Media Security Protocols

The device supports interworking media security protocols for SRTP, between SIP entities. Employing IP Profiles, you can configure the security protocol (SDES and DTLS) per SIP entity, using the IPProfile_SBCMediaSecurityMethod parameter (see Configuring IP Profiles). For more information on SDES and DTLS, see Configuring Media (SRTP) Security.

Interworking ICE Lite for NAT Traversal

The device supports interworking ICE for NAT traversal, between SIP entities. Employing IP Profiles, you can enable ICE Lite per SIP entity, using the [IPProfile_SBCIceMode] parameter (see Configuring IP Profiles).

Fax Negotiation and Transcoding

The device can allow fax transmissions to traverse transparently without transcoding or it can handle the fax as follows:

- Allow interoperability between different fax machines, supporting fax transcoding if required.
- Restrict usage of specific fax coders to save bandwidth, enhance performance, or comply with supported coders. These coders include G.711 (A-Law or Mu-Law), VBD (G.711 A-Law or G.711 Mu-Law), and T38.

Fax configuration is done in the Coder Groups table and IP Profiles table. The Coder Groups table defines the supported coders, which is assigned to the IP Profile associated with the SIP entity. The IP Profiles table also defines the negotiation method used between the incoming and outgoing fax legs, using the following fax-related parameters:

- IPProfile_SBCFaxBehavior: defines the offer negotiation method - pass fax transparently, negotiate fax according to fax settings in IP Profile, or enforce remote UA to first establish a voice channel before fax negotiation.
- IPProfile_SBCFaxCodersGroupName: defines the supported fax coders (from the Coder Groups table).
- IPProfile_SBCFaxOfferMode: determines the fax coders sent in the outgoing SDP offer.
- IPProfile_SBCFaxAnswerMode: determines the fax coders sent in the outgoing SDP answer.
- IPProfile_SBCRemoteRenegotiateOnFaxDetection: You can also configure the device to detect for faxes (CNG tone) immediately after the establishment of a voice channel (i.e., after 200 OK) and within a user-defined interval. If detected, it can then handle the subsequent fax renegotiation by sending re-INVITE messages to both SIP entities (originating and terminating faxes). For more information, see the parameter in Configuring IP Profiles.

The voice-related coder configuration (Allowed and Extension coders) is independent of the fax-related coder configuration, with the exception of the G.711 coder. If the G.711 coder is restricted by the Allowed Audio Coders Groups table, it is not used for fax processing even if it is listed in the Coder Groups table for faxes. However, support for G.711 coders for voice is not dependent upon which fax coders are listed in the Coder Groups table.

SBC Authentication

The device can authenticate SIP servers and SBC users (clients). The different authentication methods are described in the subsequent subsections.

SIP Authentication Server Functionality

The device can function as an Authentication server for authenticating received SIP message requests, based on HTTP authentication Digest with MD5. Alternatively, such requests can be authenticated by an external, third-party server.

When functioning as an Authentication server, the device can authenticate the following SIP entities:

- **SIP servers:** This is applicable to Server-type IP Groups. This provides protection from rogue SIP servers, preventing unauthorized usage of device resources and functionality. To authenticate remote servers, the device challenges the server with a user-defined username and password that is shared with the remote server. When the device receives an INVITE request from the remote server, it challenges the server by replying with a SIP 401 Unauthorized response containing the WWW-Authenticate header. The remote server then re-sends the INVITE containing an Authorization header with authentication information based on this username-password combination to confirm its identity. The device uses the username and password to authenticate the message prior to processing it.

- **SIP clients:** These are clients belonging to a User-type IP Group. This support prevents unauthorized usage of the device's resources by rogue SIP clients. When the device receives an INVITE or REGISTER request from a client (e.g., SIP phone) for SIP message authorization, the device processes the authorization as follows:

  a. The device challenges the received SIP message only if it is configured as a SIP method (e.g., INVITE) for authorization. This is configured in the IP Groups table, using the 'Authentication Method List' parameter.

  b. If the message is received without a SIP Authorization header, the device "challenges" the client by sending a SIP 401 or 407 response. The client then resends the request with an Authorization header (containing the user name and password).

  c. The device validates the SIP message according to the AuthNonceDuration, AuthChallengeMethod and AuthQOP parameters.

     - If validation fails, the device rejects the message and sends a 403 (Forbidden) response to the client.
     - If validation succeeds, the device verifies client identification. It checks that the username and password received from the client is the same username and password in the device's User Information table / database (see SBC User Information for SBC User Database). If the client is not successfully authenticated after three attempts, the device sends a SIP 403 (Forbidden) response to the client. If the user is successfully identified, the device accepts the SIP message request.

The device's Authentication server functionality is configured per IP Group, using the 'Authentication Mode' parameter in the IP Groups table (see Configuring IP Groups).

RADIUS-based User Authentication

The device can authenticate SIP clients (users) using a remote RADIUS server. The device supports the RADIUS extension for digest authentication of SIP clients, according to draft-sterman-aaa-sip-01. Based on this standard, the device generates the nonce (in contrast to RFC 5090, where it is done by the RADIUS server).

RADIUS based on draft-sterman-aaa-sip-01 operates as follows:

1. The device receives a SIP request without an Authorization header from the SIP client.
2. The device generates the nonce and sends it to the client in a SIP 407 (Proxy Authentication Required) response.
3. The SIP client sends the SIP request with the Authorization header to the device.
4. The device sends an Access-Request message to the RADIUS server.
5. The RADIUS server verifies the client's credentials and sends an Access-Accept (or Access-Reject) response to the device.
6. The device accepts the SIP client's request (sends a SIP 200 OK or forwards the authenticated request) or rejects it (sends another SIP 407 to the SIP client).

To configure this feature, set the SBCServerAuthMode ini file parameter to 2.

OAuth2-based User Authentication

The device supports the OAuth 2.0 authentication protocol (RFC 7662 and Internet Draft "draft-ietf-sipcore-sip-authn-02"), allowing it to authenticate any specified incoming SIP request (e.g., REGISTER and INVITE) with a third-party OAuth Authorization server over HTTP/S.

OAuth authorization consists of the following main stages:

1. (This stage does not involve the device.) The client application requires an OAuth Access Token for the user. There are multiple schemes to do this. For example, it may use the Authorization Code method, whereby the client application refers the user to the OAuth Authorization server to request an Authorization Code. The client application then uses the received Authorization Code to request an Access Token (and a Refresh Token) for the user from the Authorization server.
2. When the user wants to register with the device or make a call, the client application (e.g., Web browser for the WebRTC application) through which the user communicates with the device, sends a SIP REGISTER or INVITE request that includes the user's Access Token in the SIP Authorization header ("Bearer" value), as shown in the following REGISTER message example:

```
REGISTER sip:server.com SIP/2.0
Via: SIP/2.0/WSS 9rihbeck4vat.invalid;branch=z9hG4bK2426139
Max-Forwards: 69
To: <sip:alice@example.com>
From: "alice" <sip:alice@example.com>;tag=mstg4hpof6
Call-ID: 0il6hahess4ndc1pdlleqj
CSeq: 1 REGISTER
```

```
Authorization: Bearer eyJhbGciOiJSUz...
...4Oq7bK5C4aWkTUu6e...
...MgkqIC50fCb3oyiYzLbbMmZ06JA
```

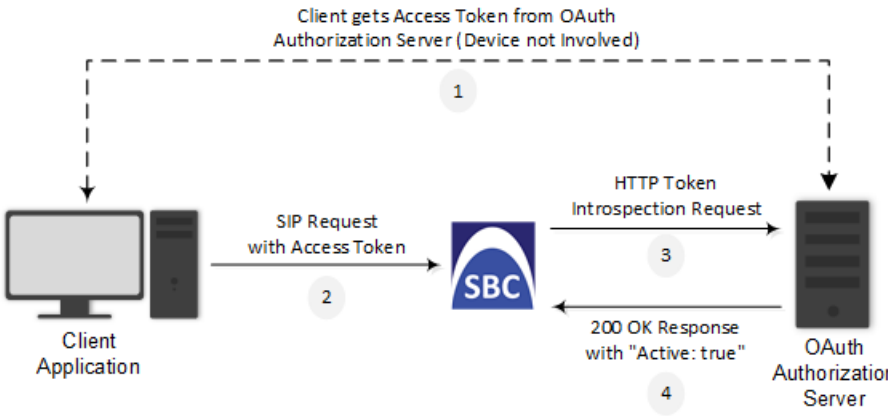
```
Contact: <sip:lnumvv6i@9rihbeck4vat.invalid;transport=ws>;+sip.ice;reg-
id=1;
+sip.instance="<urn:uuid:1007ed30-98a3-492e-966f
67b6f6eb99c5>";expires=600
Expires: 600
Allow:
INVITE,ACK,CANCEL,BYE,UPDATE,MESSAGE,OPTIONS,REFER,INFO
Supported: path,gruu,outbound
User-Agent: Example WebRTC phone
Content-Length: 0
```

3. The device authenticates the SIP request, by sending (HTTP POST) an HTTP Introspection request with the user's Access Token to the OAuth Authorization server, as shown in the following example:

```
POST /auth/realms/demo/protocol/openid-connect/token/introspect HTTP/1.1
Host: authorizationhost.com
Content-Type: application/x-www-form-urlencoded
Content-Length:…
Authorization: Basic
dGVzdEludHJvc3BlY3Q6NTliZDA4NGUtMTJlNi00N2I5LWJmNz
token=<Access Token from Bearer in SIP Authorization header>
```

4. The OAuth Authorization server checks (*introspects*) if the token is currently active (or if it has expired or revoked). Upon a successful introspection, the OAuth Authorization server sends to the device a 200 OK response containing a JSON body ("application/ json").
5. The device checks the following attributes in the received JSON body:
   - "active": A "true" value indicates a valid token and the device allows the user access to its resources and continues with the regular handling and processing of the SIP request (e.g., registers user or processes the call). A "false" value indicates an invalid token and the device responds to the SIP request with a 401 (Unauthorized) response containing the header 'WWW-Authenticate: Bearer error="invalid-token"', indicating authentication failure.
   - "username": (Optional attribute) When it exists, the device compares it to the AOR of the SIP message. For REGISTER requests, the AOR is taken from the To header; for all other requests, the AOR is taken from the From header. If the username includes a "@" character, the entire AOR is compared; otherwise, only the user-part of the AOR is compared. If comparison fails, the device responds to the SIP request with a 401 (Unauthorized) response containing the header 'WWW-Authenticate: Bearer error="invalid_request"', indicating authentication failure.

Figure 22-1: General Stages of OAuth-based Authentication



The main configuration required for OAuth-based authentication, includes the following:

- Configuring a Remote Web Service to represent the OAuth Authentication server
- Configuring the source IP Group (client) to authenticate by an OAuth Authorization server

The following provides a step-by-step example of configuring OAuth authentication.

➢ **To configure OAuth-based authentication:**

1. Open the Remote Web Services table (see Configuring Remote Web Services on page 316), and then configure a Remote Web Service to represent the OAuth Authentication server:

Parameter	Value	Comment
'Name'	"OAUth-Server"	Any descriptive name.
'Type'	**General**	-
'Path'	"auth/realms/demo/protocol/openid-connect/token/introspect"	Relative URL for the introspection service on the server.
'Username'	"device234"	Username that the device uses for authenticating the HTTP POST introspection request which it sends to the OAuth server.
'Password'	"12abMt"	Password that the device uses for authenticating the HTTP POST introspection request which it sends to the OAuth server. **Note:** The password cannot be configured with wide characters.

2. Select the Remote Web Service that you configured in Step 1, click the **HTTP Remote Hosts** link located below the table, and then configure an HTTP Remote Host:

Parameter	Value	Comment
'Address'	"oauth.example.com"	Address of the Authentication server.
'Port'	"443"	Port number of the Authentication server.

Parameter	Value	Comment
'Transport Type'	**HTTPS**	Secured HTTP.

3. Configure OAuth-based authentication for the source IP Group (client that the device needs to authenticate):

Parameter	Value	Comment
'Authentication Mode'	**SBC as Server**	The device authenticates as a server.
'Authentication Method List'	"register/setup-invite"	The SIP methods that the device needs to authenticate.
'SBC Server Authentication Type'	**Authenticate with OAuth Server**	The device authenticates the SIP requests with an OAuth Authentication server.
'OAuth HTTP Service'	**OAuth-Server**	Assigns the Remote Web Service that you configured (in Step 1) for the OAuth Authentication server.

Interworking SIP Signaling

The device supports interworking of SIP signaling messages to ensure interoperability between communicating SIP UAs or entities. This is critical in network environments where the UAs on opposing SBC legs have different SIP signaling support. For example, some UAs may support different versions of a SIP method while others may not event support a specific SIP method. The configuration method for assigning specific SIP message handling modes to UAs, includes configuring an IP Profile with the required interworking mode, and then assigning the IP Profile to the relevant IP Group.

This section describes some of the device's support for handling SIP methods to ensure interoperability.

Interworking SIP 3xx Redirect Responses

The device supports interworking of SIP 3xx redirect responses. By default, the device's handling of SIP 3xx responses is to send the Contact header unchanged. However, some SIP UAs may support different versions of the SIP 3xx standard while others may not even support SIP 3xx.

The handling of SIP 3xx can be configured for all calls, using the global parameter SBC3xxBehavior. To configure different SIP 3xx handling options for different UAs (i.e., per IP Group), use the IP Profiles table parameter, 'SBC Remote 3xx Mode'.

Resultant INVITE Traversing Device

The device can handle SIP 3xx responses so that the new INVITE message sent as a result of the 3xx traverses the device. The reasons for enforcing resultant INVITEs to traverse the device may vary:

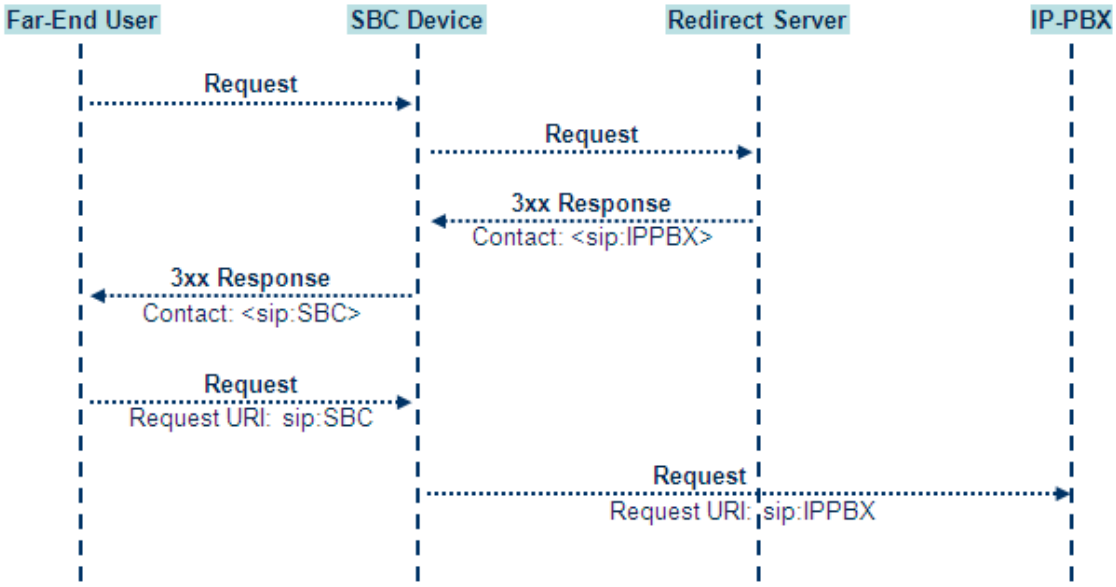
- The user that receives the 3xx is unable to route to the 3xx contact (i.e., the user is on the LAN and the new contact is on the WAN). In such a scenario, the device enables the user to reach the WAN contact and overcome NAT problems.
- Enforce certain SBC policies (e.g., call admission control, header manipulation, and transcoding) on the resultant INVITE.

The device enforces this by modifying each Contact in the 3xx response as follows:

- Changes the host part to the device's IP address – this change causes the remote user agent to send the INVITE to the device.
- Adds a special prefix ("T~&R_") to the Contact user part – to identify the new INVITE as a 3xx resultant INVITE.

The SBC handling for the 3xx resultant INVITE is as follows:

1. The incoming INVITE is identified as a 3xx resultant INVITE according to the special prefix.
2. The device automatically replaces the SBC host part (in the Request-URI) with the host from the 3xx Contact.
3. The prefix ("T~&R_") remains in the user part for the classification, manipulation, and routing mechanisms.
4. The classification, manipulation, and routing processes are done exactly like any other INVITE handling. The special prefix can be used for specific routing rules for 3xx resultant INVITEs.
5. The prefix is removed before the resultant INVITE is sent to the destination.



The process of this feature is described using an example:

1. The device receives the Redirect server's SIP 3xx response (e.g., Contact: <sip:User@IPPBX:5060;transport=tcp;param=a>;q=0.5).
2. The device replaces the Contact header value with the special prefix and database key value as user part, and with the device's URL as host part (e.g., Contact: <sip:Prefix_Key_User@SBC:5070;transport=udp>;q=0.5).
3. The device sends this manipulated SIP 3xx response to the Far-End User (FEU).
4. The FEU sends a new request with the Request-URI set to the value of the received 3xx response's Contact header (e.g., RequestURI: sip:Prefix_Key_User@SBC:5070;transport=udp).
5. Upon receipt of the new request from the FEU, the device replaces the Request-URI with the new destination address (e.g., RequestURI: sip:Prefix_User@IPPBX:5070;transport=tcp;param=a).
6. The device removes the user prefix from the Request-URI, and then sends this Request-URI to the new destination (e.g., RequestURI: sip:User@IPPBX:5070;transport=tcp;param=a).

Local Handling of SIP 3xx

The device can handle SIP 3xx responses on behalf of the dialog-initiating UA and retry the request (e.g., INVITE) using one or more alternative URIs included in the 3xx response. The new request includes SIP headers from the initial request such as Diversion, History-Info, P-Asserted-Id, and Priority. The source and destination URIs can be manipulated using the regular manipulation mechanism.

The device sends the new request to the alternative destination according to the IP-to-IP Routing table rules. (where the 'Call Trigger' field is set to **3xx**). It is also possible to specify the IP Group that sent the 3xx request as matching criteria for the re-routing rule in this table ('ReRoute IP Group ID' field).

Interworking SIP Diversion and History-Info Headers

This device can be configured to interwork between the SIP Diversion and History-Info headers. This is important, for example, to networks that support the Diversion header but not the History-Info header, or vice versa. Therefore, mapping between these headers is crucial for preserving the information in the SIP dialog regarding how and why (e.g., call redirection) the call arrived at a certain SIP UA. If the Diversion header is used, you can specify the URI type (e.g., "tel:") to use in the header, using the SBCDiversionUriType parameter.

This feature is configured in the IP Profiles table using the following parameters:

- 'Diversion Header Mode' (IPProfile_SBCDiversionMode) - defines the device's handling of the Diversion header
- 'History-Info Header Mode' (IPProfile_SBCHistoryInfoMode - defines the device's handling of the History-Info header

The handling of the SIP Diversion and History-Info headers is described in the table below:

Table 22-1: Handling of SIP Diversion and History-Info Headers

Parameter Value	SIP Header Present in incoming SIP Message		Device Action	IP Header Present in outgoing SIP Message	
	Diversion	**History-Info**		**Diversion**	**History-Info**
'Diversion Header Mode' = **Add** 'History-Info Header Mode' = **Add**	Not present	Present	Diversion added from History-Info	Present	Present
'Diversion Header Mode' = **Add** 'History-Info Header Mode' = **Add**	Present	Not present	History-Info added from Diversion	Present	Present
'Diversion Header Mode' = **Add** 'History-Info Header Mode' = **Add**	Present	Present	Diversion replaced and added from History-Info History-Info replaced and added from Diversion	Present	Present
'Diversion Header Mode' = * 'History-Info Header Mode' = *	Not present	Not present	As no headers are present on incoming message, nothing is added	Not present	Not present
'Diversion Header Mode' = **Add** 'History-Info Header	Not present	Present	Diversion added from History-Info	Present	Present

Parameter Value	SIP Header Present in incoming SIP Message		Device Action	IP Header Present in outgoing SIP Message	
Mode' = **As Is**					
'Diversion Header Mode' = **As Is** 'History-Info Header Mode' = **Add**	Present	Not present	History-Info added from Diversion	Present	Present
'Diversion Header Mode' = **Add** 'History-Info Header Mode' = **Remove**	Not present	Present	Diversion added from History-Info History-Info removed	Present	Not present
'Diversion Header Mode' = **Remove** 'History-Info Header Mode' = **Add**	Present	Not present	History-Info added from Diversion Diversion removed	Not present	Present
'Diversion Header Mode' = **Remove** 'History-Info Header Mode' = **Remove**	Present	Present	Both removed	Not present	Not present

Interworking SIP REFER Messages

The device supports interworking of SIP REFER messages. SIP UAs may support different versions of the REFER standard while others may not even support REFER.

This feature supports the following:

- Attended, unattended, and semi-attended call transfers
- Sending INVITE, REFER-notifications, BYE, PRACK and Session Timer on behalf of peer PBXs

- Advanced routing rules for the new, initiated INVITE
- Forwarding early media after REFER while attempting to avoid transcoding (by sending session update)
- Interoperate with environments were different SIP UAs lack basic SIP functionality such as re-INVITE, UPDATE, PRACK, Delayed Offer, re-INVITE without SDP
- Session updates after connect to avoid transcoding

The handling of REFER can be configured for all calls, using the global parameter [SBCReferBehavior]. To configure different REFER handling options for different UAs (i.e., IP Groups), use the IP Profiles table parameter, 'Remote REFER Mode'.

- Local handling of REFER: This option is used for UAs that do not support REFER. Upon receipt of a REFER request, instead of forwarding it to the IP Group, the device handles it locally. It generates a new INVITE to the alternative destination according to the rules in the IP-to-IP Routing table (where the 'Call Trigger' field is set to **REFER**). It is also possible to specify the IP Group that sent the REFER request, as matching criteria for the re-routing rule in this table ('ReRoute IP Group ID' field).
- Transparent handling: The device forwards the REFER with the Refer-To header unchanged.
- Re-routing through SBC: The device changes the Refer-To header so that the re-routed INVITE is sent through the SBC application.
- IP Group Name: The device sets the host part in the REFER message to the name configured for the IP Group in the IP Groups table.

Interworking SIP PRACK Messages

The device supports interworking of SIP Provisional Response ACKnowledgement (PRACK) messages (18x). While some UAs may not support PRACK (RFC 3262) others may require it. The device can be configured to resolve this interoperable issue and enable sessions between such endpoints. SIP PRACK handling is configured using the IP Profile parameter, 'SBC Prack Mode':

- Optional: PRACK is optional for these UAs. If required, the device performs the PRACK process on behalf of the destination UA.
- Mandatory: PRACK is required for these UAs. Calls from UAs that do not support PRACK are rejected. Calls destined to these UAs are also required to support PRACK.
- Transparent (default): The device does not intervene with the PRACK process and forwards the request as is.

Interworking SIP Session Timer

The device supports interworking of the SIP signaling keep-alive mechanism. The SIP standard provides a signaling keep-alive mechanism using re-INVITE and UPDATE messages. In certain setups, keep-alive may be required by some SIP UAs while for others it may not be supported. The device can resolve this mismatch by performing the keep-alive process on behalf of SIP UAs that do not support it.

To configure the handling of session expires, use the IP Profile parameter, 'SBC Session Expires Mode'.

Interworking SIP Early Media

The device supports early media. Early media is when the media flow starts before the SIP call is established (i.e., before the 200 OK response). This occurs when the first SDP offer-answer transaction completes. The offer-answer options can be included in the following SIP messages:

- Offer in first INVITE, answer on 180, and no or same answer in the 200 OK
- Offer in first INVITE, answer on 180, and a different answer in the 200 OK (not standard)
- INVITE without SDP, offer in 180, and answer in PRACK
- PRACK and UPDATE transactions can also be used for initiating subsequent offer-answer transactions before the INVITE 200 OK response.
- In a SIP dialog life time, media characteristics after originally determined by the first offer-answer transaction can be changed by using subsequent offer-answer transactions. These transactions may be carried either in UPDATE or re-INVITE transactions. The media handling is similar to the original offer-answer handling. If the offer is rejected by the remote party, no media changes occur (e.g., INVITE without SDP, then 200 OK and ACK, offer-answer within an offer-answer, and Hold re-INVITE with IP address of 0.0.0.0 - IP address is unchanged).

The device supports various interworking modes for early media between SIP UAs (i.e., IP Groups):

- **Early Media Enabling:** The device supports the interworking of early media between SIP UAs that support early media and those that do not support receipt of early media. Early media can arrive in provisional responses to an INVITE request. The device forwards the request of early media for IP Groups that support this capability; otherwise, the device terminates it. Provisional responses whose SDP are suppressed are changed to a SIP 180 response. This feature is also supported for delayed offers. This is configured using the IP Profile parameter, 'SBC Remote Early Media Support'. The device refers to the parameter also for features that require early media such as playing ringback tone.
- **Early Media Response Type:** The device supports the interworking of different SIP provisional response types between UAs for forwarding the early media to the caller. This can support all early media response types (default), SIP 180 only, or SIP 183 only, and is configured by the IP Profile parameter, 'SBC Remote Early Media Response Type'.
- **Multiple 18x:** The device supports the interworking of different support for multiple 18x responses (including 180 Ringing, 181 Call is Being Forwarded, 182 Call Queued, and 183 Session Progress) that are forwarded to the caller. The UA can be configured as supporting only receipt of the first 18x response (i.e., the device forwards only this response to the caller), or receipt of multiple 18x responses (default). This is configured by the IP Profile parameter, 'SBC Remote Multiple 18x Support'.

- **Early Media RTP:** The device supports the interworking with remote clients that send 18x responses with early media and whose subsequent RTP is delayed, and with remote clients that do not support this and require RTP to immediately follow the 18x response. Some clients do not support 18x with early media, while others require 18x with early media (i.e., they cannot play ringback tone locally). These various interworking capabilities are configured by the IP Profile parameters, 'Remote Early Media RTP Detection Mode', 'SBC Remote Supports RFC 3960', and 'SBC Remote Can Play Ringback'. See the flowcharts below for the device's handling of such scenarios:

Figure 22-2: SBC Early Media RTP - 18x without SDP

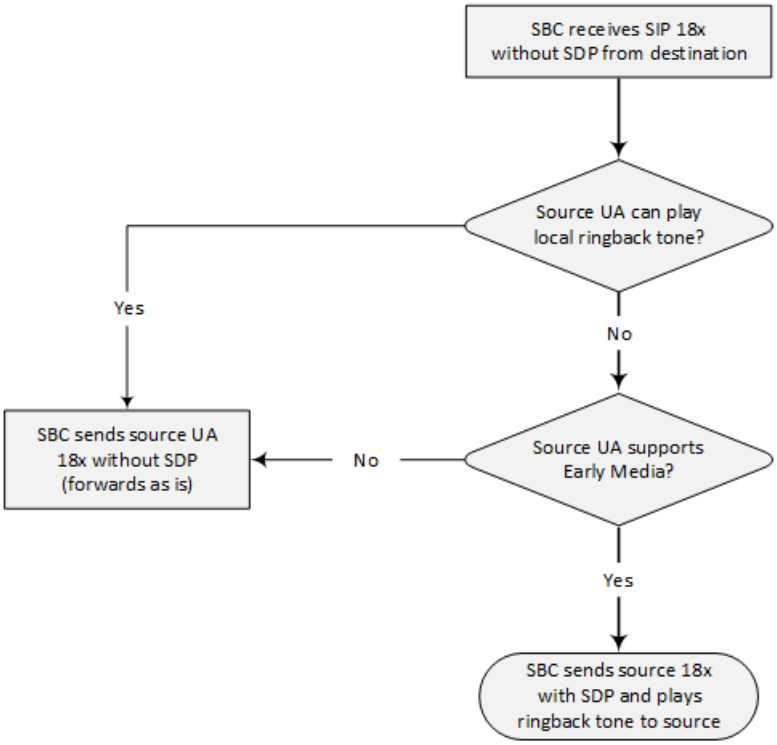
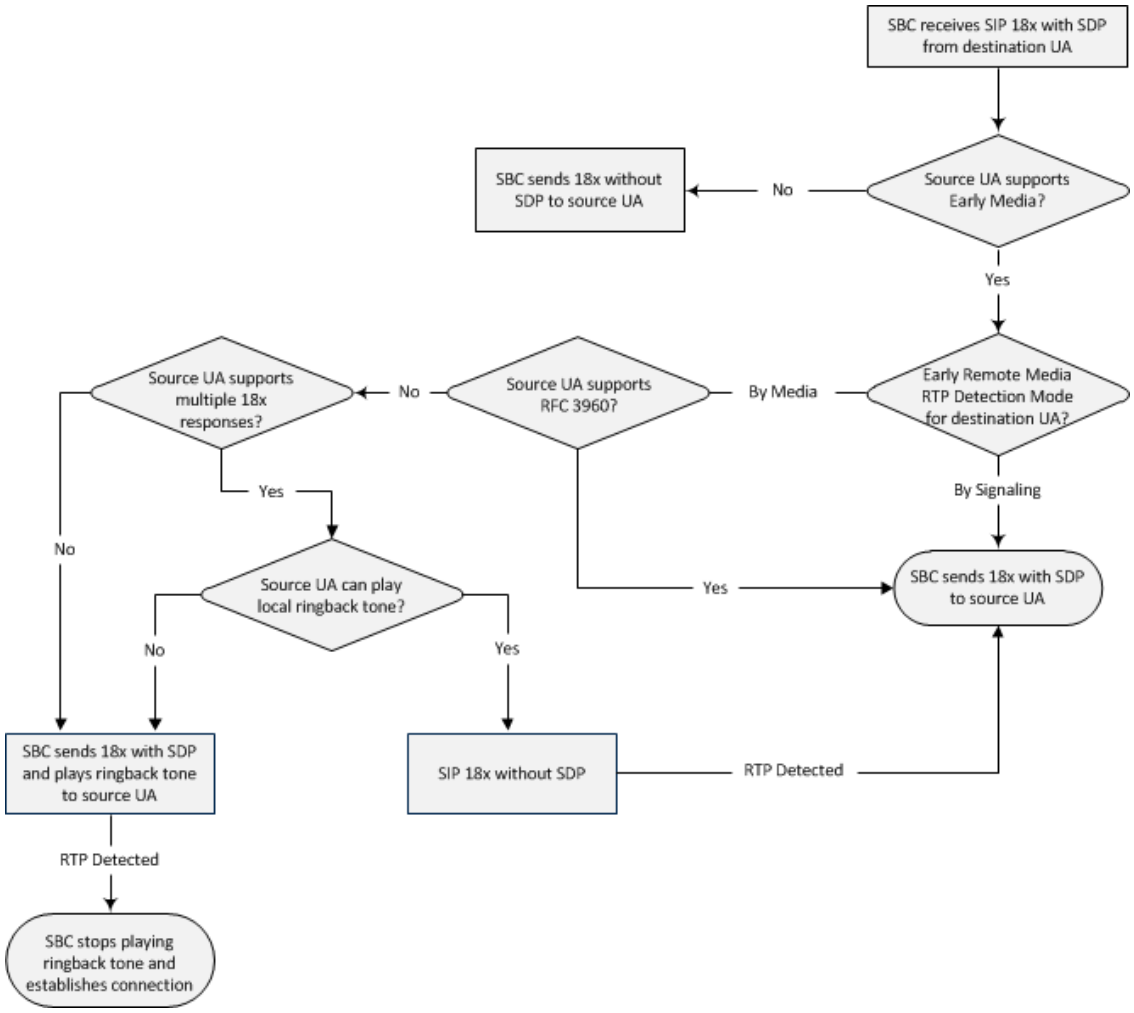


Figure 22-3: SBC Early Media RTP - 18x with SDP



Interworking SIP re-INVITE Messages

The device supports interworking of SIP re-INVITE messages. This enables communication between endpoints that generate re-INVITE requests and those that do not support the receipt of re-INVITEs. The device does not forward re-INVITE requests to IP Groups that do not support it. Instead, it sends a SIP response to the re-INVITE request, which can either be a success or a failure, depending on whether the device can bridge the media between the endpoints. The device can also handle re-INVITEs with or without an SDP body, enabling communication between endpoints that do not support re-INVITE requests without SDP, and those that require SDP. The device generates an SDP offer and adds it to the incoming re-INVITE request if it does not contain an SDP and only then forwards it to the destination endpoint. This interworking support is configured by the IP Profile parameter, 'SBC Remote Reinvite Support'.

Interworking SIP UPDATE Messages

The device supports interworking of the SIP UPDATED message. This enables communication between UAs that generate UPDATE requests and those that do not support the receipt of UPDATE requests. The device does not forward UPDATE requests to IP Groups that do not

support it. Instead, it sends a SIP response to the UPDATE request which can either be a success or a failure, depending on whether the device can bridge the media between the endpoints. The handling of UPDATE messages is configured by the IP Profile parameter 'SIP UPDATE Support'.

Interworking SIP re-INVITE to UPDATE

The device enables communication between endpoints (IP Groups) that do not support re-INVITE requests but support the UPDATE method, and vice versa. The device translates the re-INVITE request to the UPDATE request, and vice versa. Note that if a re-INVITE request arrives without SDP, the device generates the SDP and adds it to the outgoing UPDATE request. To enable this feature, each IP Group needs to be configured with its unique capabilities by associating it with a relevant IP Profile. For example, an IP Group that supports UPDATE requests but not re-INVITEs would be configured as follows:

- SBCRemoteUpdateSupport = 2 (Supported)
- SBCRemoteReinviteSupport = 0 (Not Supported)

If a re-INVITE request needs to be forwarded to this IP Group, it is translated to an UPDATE request.

Interworking Delayed Offer

The device supports interworking of INVITE messages with and without SDP between SIP entities. The device enables sessions between endpoints (IP Groups) that send INVITEs without SDP (i.e., delayed media) and those that do not support the receipt of INVITEs without SDP. The device creates an SDP and adds it to INVITEs that arrive without SDP. Delayed offer is also supported when early media is present.

Employing IP Profiles, you can configure this interworking feature per SIP entity, using the 'SBC Remote Delayed Offer Support' parameter (see Configuring IP Profiles).

- The above mentioned intervention in the SDP offer-answer process may require transcoding.
- For SIP entities that do not support delayed offer, you must assign extension coders to its IP Profile (using the 'Extension Coders' parameter).

Interworking Call Hold

The device supports the interworking of call hold / retrieve requests between SIP entities supporting different call hold capabilities:

- Interworking SDP call hold formats. This is configured by the IP Profile parameter, 'SBC Remote Hold Format'.
- Interworking the play of the held tone for IP entities that cannot play held tones locally. This is configured by the IP Profile parameter, 'Play Held Tone'.

- Interworking generation of held tone where the device generates the tone to the held party instead of the call hold initiator. This is configured by the IP Profile parameter, 'SBC Reliable Held Tone Source'.

To configure IP Profiles, see Configuring IP Profiles.

Interworking SIP Via Headers

The device supports the interworking of SIP Via headers between SIP entities. For the outgoing message sent to a SIP entity, the device can remove or retain all the Via headers received in the incoming SIP request from the other side. Employing IP Profiles, you can configure this interworking feature per SIP entity, using the IpProfile_SBCKeepVIAHeaders parameter (see Configuring IP Profiles).

Interworking SIP User-Agent Headers

The device supports the interworking of SIP User-Agent headers between SIP entities. For the outgoing message sent to a SIP entity, the device can remove or retain all the User-Agent headers received in the incoming SIP request/response from the other side. Employing IP Profiles, you can configure this interworking feature per SIP entity, using the IpProfile_ SBCKeepUserAgentHeader parameter (see Configuring IP Profiles).

Interworking SIP Record-Route Headers

The device supports the interworking of SIP Record-Route headers between IP entities. For the outgoing message sent to a SIP entity, the device can remove or retain all the Record-Route headers received in the incoming SIP request/response from the other side. Employing IP Profiles, you can configure this interworking feature per SIP entity, using the IpProfile_ SBCKeepRoutingHeaders parameter (see Configuring IP Profiles).

Interworking SIP To-Header Tags in Multiple SDP Answers

The device supports the interworking of SIP To-header tags in call forking responses (i.e., multiple SDP answers) between IP entities. The device can either use the same To-header tag value for all SDP answers sent to the SIP entity, or send each SDP answer with its original tag. Employing IP Profiles, you can configure this interworking feature per SIP entity, using the IpProfile_SBCRemoteMultipleEarlyDialogs parameter (see Configuring IP Profiles).

Interworking In-dialog SIP Contact and Record-Route Headers

The device supports the interworking of in-dialog, SIP Contact and Record-Route headers between SIP entities. Employing IP Profiles, you can configure this interworking feature per SIP entity, using the IpProfile_SBCRemoteRepresentationMode parameter (see Configuring IP Profiles).

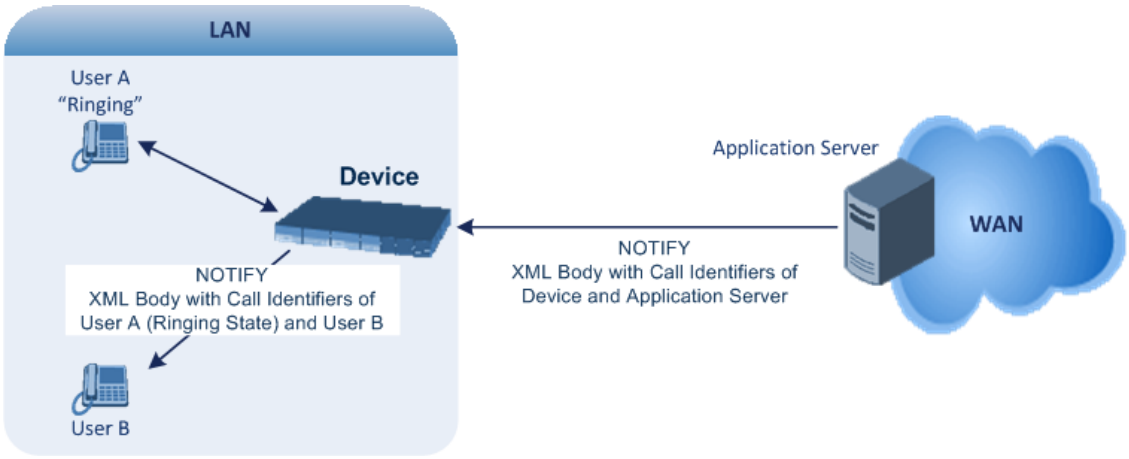
23 Configuring General SBC Settings

This section describes configuration of various SBC features.

Interworking Dialog Information in SIP NOTIFY Messages

You can enable the device to interwork dialog information (XML body) received in SIP NOTIFY messages from a remote (WAN) application server. The NOTIFY message is sent by application servers to notify a SIP client, subscribed to a service and located behind the device (LAN), of the status of another SIP client in the LAN. For example, user B can subscribe to an application server for call pick-up service, whereby if user A's phone rings, the application server notifies user B. User B can then press a pre-configured key sequence to answer the call.

The NOTIFY message contains the XML body with call identifiers (call-id and tags). However, as the application server is located in the external network WAN and the SIP clients behind the device, the call dialog information sent by the application server reflects only the dialog between the device and itself; not that of the involved SIP clients. This is due to, for example, the device's topology hiding (e.g., IP address) of its LAN elements. The device resolves this by replacing the call identifiers received from the application server with the correct call identifiers (e.g., user A and user B). Thus, users subscribed to the service can receive relevant NOTIFY messages from the device and use the service.



➢ **To enable the feature:**

- Configure the 'SBC Dialog-Info Interworking' (EnableSBCDialogInfoInterworking) parameter to **Enable**.

When the feature is disabled, the device forwards the NOTIFY message as is, without modifying its XML body.

Below is an example of an XML body where the call-id, tags, and URIs have been replaced by the device:

```
<?xml version="1.0"?>
<dialog-info xmlns="urn:ietf:params:xml:ns:dialog-info"
```

```
version="10" state="partial"
entity="sip:alice@example.com">
<dialog id="zxcvbnm3" call-id="67402270@10.132.10.150"
local-tag="1c137249965"
remote-tag="CCDORRTDRKIKWFVBRWYM" direction="initiator">
<state event="replaced">terminated</state>
</dialog>
<dialog id="sfhjsjk12" call-id="67402270@10.132.10.150"
local-tag="1c137249965"
remote-tag="CCDORRTDRKIKWFVBRWYM" direction="receiver">
<state reason="replaced">confirmed</state>
<replaces
call-id="67402270@10.132.10.150"
local-tag="1c137249965"
remote-tag="CCDORRTDRKIKWFVBRWYM"/>
<referred-by>
sip:bob-is-not-here@vm.example.net
</referred-by>
<local>
<identity display="Jason Forster">
sip:jforsters@home.net
</identity>
<target uri="sip:alice@pc33.example.com">
<param pname="+sip.rendering" pval="yes"/>
</target>
</local>
<remote>
<identity display="Cathy Jones">
sip:cjones@example.net
</identity>
<target uri="sip:line3@host3.example.net">
<param pname="actor" pval="attendant"/>
<param pname="automaton" pval="false"/>
</target>
</remote>
</dialog>
</dialog-info>
```

24 Configuring Call Admission Control

You can implement Call Admission Control (CAC) to regulate the volume of voice traffic handled by the device.

CAC configuration is done using two tables with parent-child type relationship:

- Call Admission Control Profile table: This is the parent table, which defines a name for the CAC profile.
- Call Admission Control Rule table: This is the child table, which defines the actual CAC rules for the profile.

You can configure up to 625 CAC profiles and up to 625 CAC rules. In addition, a CAC profile can be configured with up to 8 CAC rules.

Once you have configured a CAC profile with CAC rules, you need to assign it to any of the following SIP configuration entities (using the 'CAC Profile' parameter):

- IP Group (see Configuring IP Groups on page 451)
- SIP Interface (see Configuring SIP Interfaces on page 434)
- SRD (see Configuring SRDs on page 421)

CAC rules define the maximum number of allowed concurrent calls (SIP dialog-initiating requests) for the assigned SIP entity (listed above) and per registered user belonging to the SIP entity. This can also include the maximum number of allowed concurrent SIP dialogs per second (*rate*). The CAC rule can be defined for a specific SIP message type (e.g., only INVITEs) as well as for a specific call direction (e.g., only outbound calls).

The CAC feature supports SIP-dialog rate control using the token-bucket mechanism. Token bucket is a control mechanism that determines the rate of SIP dialog processing based on the presence of tokens in the bucket. Tokens in the bucket are removed ("cashed in") for the ability to process each dialog. If there are no tokens, the device rejects the dialog request with a SIP 480 (Temporarily Unavailable). Configuration of the token-bucket mechanism involves the following:

- Configuring the number of tokens that are added to the bucket per second. This is referred to as *rate*. To process (allow) a SIP dialog, the device needs a token from the bucket.
- Configuring the maximum number of tokens that the bucket can hold and thus, the maximum number of tokens that can be used for processing SIP dialogs that are received at one time. This is referred to as *burst*.

For example, assume that the rate is configured to 1 and the burst to 4:

- One token is added to the bucket every second.
- The maximum number of tokens that the bucket can hold is four.
- If SIP dialogs have never been received by the device, the bucket is filled to its maximum, which is four tokens (i.e., burst), regardless of the number of seconds that have passed.

- If four SIP dialogs are received at the same time (i.e., burst), the device uses the four tokens to process the dialogs. The bucket is now left with no tokens at that given moment, but after a second, a new token is added to the bucket (due to the rate). If there are no calls for the next three seconds, the bucket fills up again to four tokens (and no more).
- If the bucket contains four tokens (i.e., full) and five SIP dialogs are received at the same time, the device uses the four tokens to process four of the dialogs and rejects one.
- If the bucket has one token and SIP dialogs are then received every second, the device uses the token to process the first dialog, adds a token to the bucket after a second and processes the second dialog, and so on.

Your CAC rule can also define a guaranteed number of concurrent calls (reserved capacity) for the assigned SIP entity (see above) . Reserved capacity is especially useful when the device operates with multiple entities. For example, if the total call capacity supported by the device is 200, a scenario may arise where a SIP entity may reach 200 call sessions, leaving no available call resources for the other SIP entities. If the reserved call capacity of a SIP entity is threatened by a new call for a different SIP entity, the device rejects the call to safeguard the reserved capacity.

Requests that reach the user-defined call limit (maximum concurrent calls or call rate) are sent to an alternative route, if configured (in the IP-to-IP Routing table). If no alternative routing rule exists, the device rejects the SIP request with a SIP 480 "Temporarily Unavailable" response.

- The device applies the CAC rule for the incoming leg immediately after the Classification process. If the call/request is rejected at this stage, no routing is performed. The enforcement for the outgoing leg is performed within each alternative route iteration. This is accessed from two places - during initial classification/routing and during alternative routing.
- CAC does not apply to Test Calls.

The following procedure describes how to configure CAC profiles through the Web interface. You can also configure them through other management interfaces:

- Call Admission Control Profile table: ini file [SBCAdmissionProfile] or CLI (`configure voip > sbc cac-profile`)
- Call Admission Control Rule table: ini file [SBCAdmissionRule] or CLI (`configure voip > sbc cac-rule`)

➢ To configure a CAC profile:

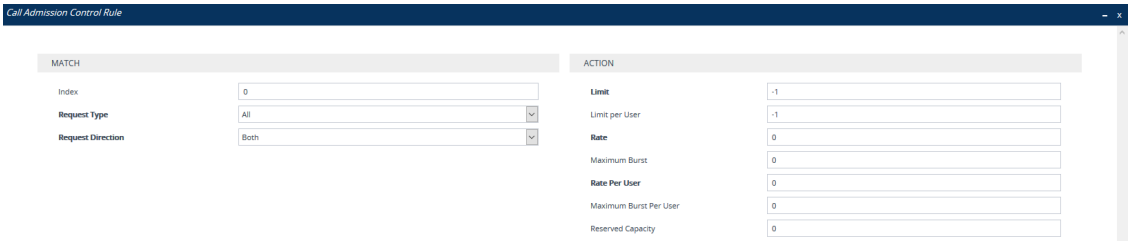
1. Open the Call Admission Control Profile table (**Setup** menu > **Signaling & Media** tab > **SBC** folder > **Call Admission Control Profile**).
2. Click **New**; the following dialog box appears:

3. Configure a CAC profile according to the parameters described in the table below.
4. Click **Apply**.

Table 24-1: Call Admission Control Profile Table Parameter Description

Parameter	Description
'Index' [SBCAdmissionProfile_Index]	Defines an index number for the new table row. **Note:** Each row must be configured with a unique index.
'Name' `name` [SBCAdmissionProfile_Name]	Defines a descriptive name, which is used when associating the row in other tables. The valid value is a string of up to 40 characters. By default, no value is defined. **Note:** The parameter value cannot contain a forward slash (/).

5. In the Call Admission Control Profile table, select the required row, and then click the **Call Admission Control Rule** link located below the table; the Call Admission Control Rule table appears.
6. Click **New**; the following dialog box appears:



7. Configure a CAC rule according to the parameters described in the table below.
8. Click **Apply**.

Table 24-2: Call Admission Control Rule Table Parameter Description

Parameter	Description
Match	
'Index' `sbc-admission-rule-`	Defines an index number for the new table row. **Note:** Each row must be configured with a unique index.

Parameter	Description
<Index>/<Index> [SBCAdmissionRule_RuleIndex]	
'Request Type' request-type [SBCAdmissionRule_RequestType]	Defines the type of SIP dialog-initiating request to which you want to apply the rule (not to subsequent requests, which can be of different type and direction). ■ [0] **All** (default) ■ [1] **INVITE** ■ [2] **SUBSCRIBE** ■ [3] **Other** = All SIP request types except INVITEs and SUBSCRIBEs (e.g., REGISTER).
'Request Direction' request-direction [SBCAdmissionRule_RequestDirection]	Defines the call direction of the SIP request to which the rule applies. ■ [0] **Both** = (Default) Rule applies to inbound and outbound SIP dialogs. ■ [1] **Inbound** = Rule applies only to inbound SIP dialogs. ■ [2] **Outbound** = Rule applies only to outbound SIP dialogs.
Action	
'Limit' limit [SBCAdmissionRule_Limit]	Defines the maximum allowed number of concurrent SIP dialogs. You can also use the following special values: ■ [-1] **-1** = (Default) Unlimited. ■ [0] **0** = Block all the SIP dialog types specified in the 'Request Type' parameter (above).
'Limit per User' limit-per-user [SBCAdmissionRule_LimitPerUser]	Defines the maximum allowed number of concurrent SIP dialogs per user. You can also use the following special values: ■ [-1] **-1** = (Default) Unlimited. ■ [0] **0** = Block all the SIP dialog types specified in the 'Request Type' parameter (above).
'Rate' rate [SBCAdmissionRule_Rate]	Defines the number of tokens added to the token "bucket" per second, where a token "buys" a SIP dialog. For example, if you configure the parameter to 1, one

Parameter	Description
	token is added to the bucket every second. If there are no calls for five seconds, the bucket would have accumulated 5 tokens. The default is 0 (i.e., unlimited rate). **Note:** If you configure this parameter, you must also configure the 'Maximum Burst' parameter to a non-zero value.
'Maximum Burst' `max-burst` [SBCAdmissionRule_MaxBurst]	Defines the maximum number of SIP dialogs that can be processed at any given time. In other words, it defines the maximum number of tokens that the "bucket" can hold. The device only accepts a SIP dialog if a token exists in the "bucket". Once the SIP dialog is accepted, a token is removed from the "bucket". If a SIP dialog is received by the device and the token "bucket" is empty, the device rejects the SIP dialog. Alternatively, if the "bucket" is full, for example, 100 tokens, and 101 SIP dialogs arrive (before another token is added to the "bucket", i.e., faster than that configured in the 'Rate' parameter), the device accepts the first 100 SIP dialogs and rejects the last one. The device sends a SIP 480 “Temporarily Unavailable” response when it rejects requests. Dropped requests are not counted in the "bucket". The default is 0 (i.e., unlimited SIP dialogs). **Note:** ■ The parameter functions together with the 'Rate' parameter (see above). ■ The parameter's value cannot be greater than 10 times (x) the value of the 'Rate' parameter. For example, if you configured the 'Rate' parameter to 2, you can configure the 'Maximum Burst' parameter to any value less than or equal to 20 (i.e., 10 x 2). ■ The token bucket feature is per SIP request type and SIP request direction.
'Rate Per User' `rate-per-user` [SBCAdmissionRule_ RatePerUser]	Defines the maximum allowed number of concurrent SIP dialogs per registered user that can be handled per second. The default is 0 (i.e., unlimited rate).

Parameter	Description
	Note: If you configure this parameter, you must also configure the 'Maximum Burst per User' parameter to a non-zero value (see below).
'Maximum Burst Per User' `max-burst-per-user` [SBCAdmissionRule_ MaxBurstPerUser]	Defines the maximum number of tokens (SIP dialogs) per user that the bucket can hold (see the 'Maximum Burst' parameter for a detailed description). The default is 0 (i.e., unlimited SIP dialogs). **Note:** The parameter functions together with the 'Rate Per User' parameter (see above).
'Reserved Capacity' `reservation` [SBCAdmissionRule_ Reservation]	Defines the guaranteed (minimum) call capacity. The default is 0 (i.e., no reserved capacity). If you configure reserved call capacity for an SRD and each of its associated IP Groups, the SRD's reserved call capacity must be greater or equal to the summation of the reserved call capacity of all these IP Groups. In other words, the SRD serves as the "parent" reserved call capacity. If the SRD's reserved call capacity is greater, the extra call capacity can be used as a shared pool between the IP Groups for unreserved calls when they exceed their reserved capacity. For example, assume that the reserved capacity for an SRD and its associated IP Groups are as follows: ■ SRD reserved call capacity: 40 ■ IP Group ID 1 reserved call capacity: 10 ■ IP Group ID 2 reserved call capacity: 20 In this setup, the SRD offers a shared pool for unreserved call capacity of 10 [i.e., 40 – (10 + 20)]. If IP Group ID 1 needs to handle 15 calls, it is guaranteed 10 calls and the remaining 5 is provided from the SRD's shared pool. If the SDR's shared pool is currently empty and resources for new calls are required, the quota is taken from the device's total capacity, if available. For example, if IP Group ID 1 needs to handle 21 calls, it's guaranteed 10, the SRD's shared pool provides another 10, and the last call is provided from the device's total call capacity support (e.g., of 200). **Note:** ■ Reserved call capacity is applicable only to IP Groups

Parameter	Description
	and SRDs. ■ Reserved call capacity is applicable only to INVITE and SUBSCRIBE messages. ■ Reserved call capacity must be less than the maximum capacity (limit) configured for the CAC rule (see the 'Limit' parameter). ■ The total reserved call capacity configured for all CAC rules must be within the device's total call capacity support.

25 Routing SBC

This section describes configuration of call routing for the SBC application.

Configuring Classification Rules

The Classification table lets you configure up to 625 Classification rules. A Classification rule classifies incoming SIP dialog-initiating requests (e.g., INVITE messages) to a "source" IP Group. The source IP Group is the SIP entity that sent the SIP dialog request. Once classified, the device uses the IP Group to process the call (manipulation and routing).

You can also use the Classification table for employing SIP-level access control for successfully classified calls, by configuring Classification rules with whitelist and blacklist settings. If a Classification rule is configured as a whitelist ("Allow"), the device accepts the SIP dialog and processes the call. If the Classification rule is configured as a blacklist ("Deny"), the device rejects the SIP dialog.

Configuration of Classification rules includes two areas:

- **Match:** Defines the matching characteristics of the incoming IP call (e.g, source SIP Interface and IP address). Classification is primarily based on the SIP Interface (as the matching characteristics) on which the incoming dialog is received. As Classification rules must first be assigned with an SRD, the SIP Interface is one that belongs to the SRD. Therefore, Classification rules are configured per SRD, where multiple SIP Interfaces can be used as matching characteristics. However, as multiple SRDs are relevant only for multi-tenant deployments, for most deployments only a single SRD is required. As the device provides a default SRD ("Default_SRD"), when only one SRD is required, the device automatically assigns it to the Classification rule.
- **Action:** Defines the action that is done if the incoming call matches the characteristics of the rule (i.e., classifies the call to the specified IP Group).

The device searches the table from **top to bottom** for the **first** rule that matches the characteristics of the incoming call. If it finds a matching rule, it classifies the call to the IP Group configured for that rule. If you are using source tags to classify incoming calls to IP Groups, then once the device locates a matching rule (including a match for the source tag), the device searches the IP Groups table for an IP Group with the matching tag. For more information on classification based on tags, see Configuring Classification Based on Tags on page 714.

Configure stricter classification rules higher up in the table than less strict rules to ensure incoming dialogs are classified to the desired IP Group. *Strict* refers to the number of matching characteristics configured for the rule. For example, a rule configured with source host name and destination host name as matching characteristics is stricter than a rule configured with only source host name. If the rule configured with only source host name appears higher up in the table, the device ("erroneously") uses the rule to classify incoming dialogs matching this source host name (even if they also match the rule appearing lower down in the table configured with the destination host name as well).

If the device doesn't find a matching rule (i.e., classification fails), the device rejects or allows the call depending on the following configuration:

➢ **To configure the action for unclassified calls:**

1. Open the SBC General Settings (**Setup** menu > **Signaling & Media** tab > **SBC** folder > **SBC General Settings**).
2. From the 'Unclassified Calls' drop-down list, select **Reject** to reject unclassified calls or **Allow** to accept unclassified calls:

3. Click **Apply**.

If you configure the parameter to **Allow**, the incoming SIP dialog is assigned to an IP Group as follows:

1. The device determines on which SIP listening port (e.g., 5061) the incoming SIP dialog request was received and the SIP Interface configured with this port (in the SIP Interfaces table).
2. The device determines the SRD associated with this SIP Interface (in the SIP Interfaces table) and then classifies the SIP dialog to the first IP Group in the IP Groups table that is associated with the SRD. For example, if IP Groups 3 and 4 belong to the same SRD, the device classifies the call to IP Group 3.

The Classification table is used to classify incoming SIP dialog requests **only if** the following classification stages **fail**:

1. **Classification Stage 1 - Based on User Registration Database:** The device searches its users registration database to check whether the incoming SIP dialog arrived from a registered user. The device searches the database for a user that matches the address-of-record (AOR) and Contact of the incoming SIP message:
   - Compares the SIP Contact header to the contact value in the database.
   - Compares the URL in the SIP P-Asserted-Identity/From header to the registered AOR in the database.

If the device finds a matching registered user, it classifies the user to the IP Group associated with the user in the database. If this classification stage fails, the device proceeds to classification based on Proxy Set.

2. **Classification Stage 2 - Based on Proxy Set:** If the database search fails, the device performs classification based on Proxy Set. This classification is applicable only to Server-type IP Groups and is done only if classification based on Proxy Set is enabled (see the 'Classify By Proxy Set' parameter in the IP Groups table in Configuring IP Groups). The device checks whether the incoming INVITE's IP address (if host name, then according to the dynamically resolved IP address list) is configured for a Proxy Set (in the Proxy Sets table). If such a Proxy Set exists, the device classifies the INVITE to the IP Group that is associated with the Proxy Set. The Proxy Set is assigned to the IP Group in the IP Groups table.

   If more than one Proxy Set is configured with the same IP address and associated with the same SIP Interface, the device may classify and route the SIP dialog to an incorrect IP Group. In such a scenario, a warning is generated in the Syslog message. However, if some Proxy Sets are configured with the same IP address but different ports (e.g., 10.1.1.1:5060 and 10.1.1.1:5070) and the 'Classification Input' parameter is configured to **IP Address, Port & Transport Type**, classification (based on IP address and port combination) to the correct IP Group is achieved. Therefore, when classification is by Proxy Set, pay attention to the configured IP addresses and the 'Classification Input' parameter of your Proxy Sets. When more than one Proxy Set is configured with the same IP address, the device selects the matching Proxy Set in the following precedence order:

   a. Selects the Proxy Set whose IP address, port, and transport type match the source of the incoming dialog.

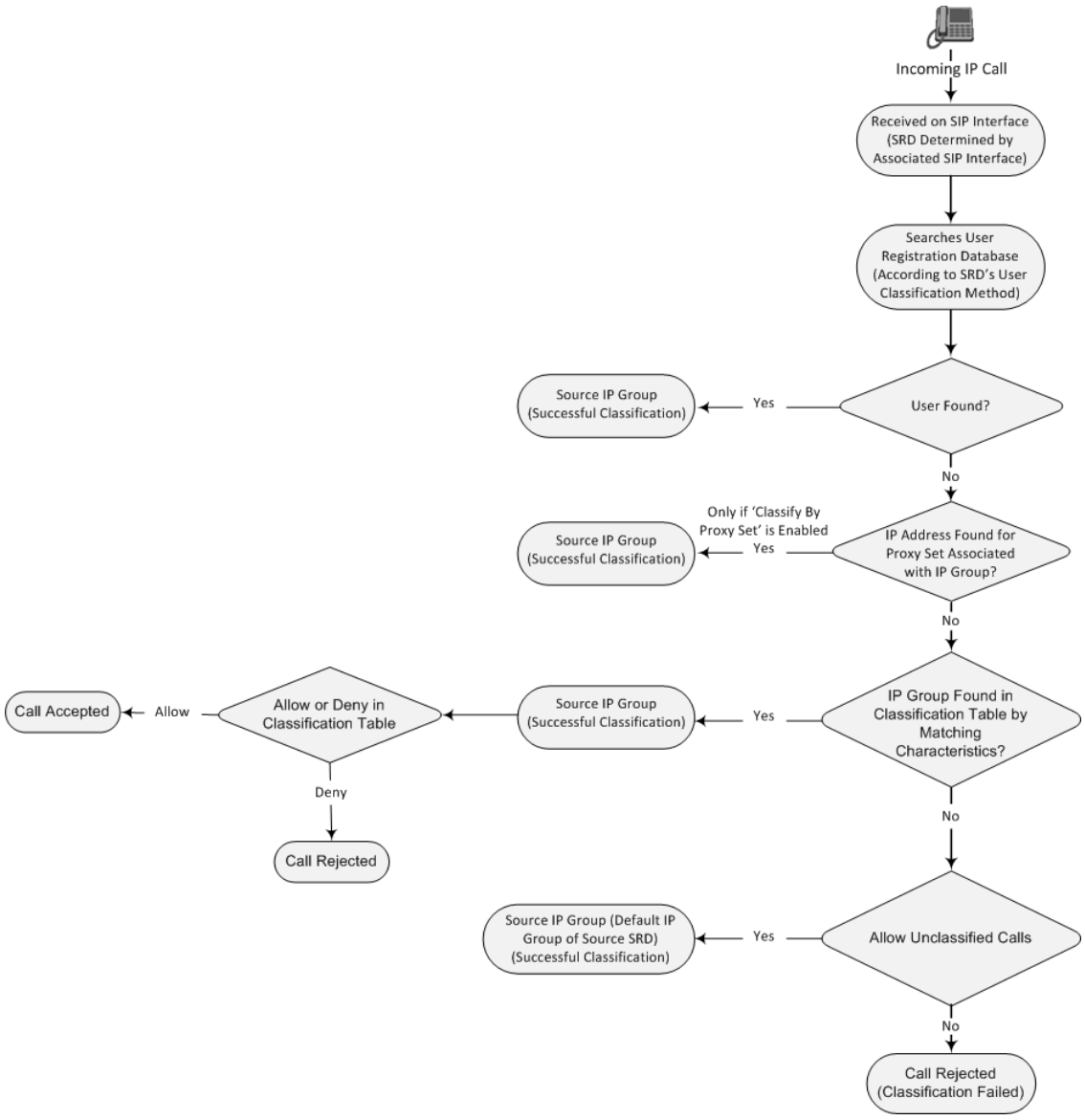
   b. If no match is found for a), it selects the Proxy Set whose IP address and transport type match the source of the incoming dialog (if the 'Classification Input' parameter is configured to **IP Address Only**).

   c. If no match is found for b), it selects the Proxy Set whose IP address match the source of the incoming dialog (if the 'Classification Input' parameter is configured to **IP Address Only**).

   If classification based on Proxy Set fails (or classification based on Proxy Set is disabled), the device proceeds to classification based on the Classification table.

- For security, it is recommended to classify SIP dialogs based on Proxy Set only if the IP address of the Server-type IP Group is **unknown**. In other words, if the Proxy Set associated with the IP Group is configured with an FQDN. In such cases, the device classifies incoming SIP dialogs to the IP Group based on the DNS-resolved IP address. If the IP address is known, it is recommended to use a Classification rule instead (and disable the Classify by Proxy Set feature), where the rule is configured with not only the **IP address, but also with SIP message characteristics** to increase the strictness of the classification process. The reason for preferring classification based on Proxy Set when the IP address is unknown is that IP address forgery (commonly known as IP spoofing) is more difficult than malicious SIP message tampering and therefore, using a Classification rule without an IP address offers a weaker form of security. When classification is based on Proxy Set, the Classification table for the specific IP Group is ignored.
- If multiple IP Groups are associated with the same Proxy Set, use Classification rules to classify the incoming dialogs to the IP Groups (do **not** use the Classify by Proxy Set feature).
- The device saves incoming SIP REGISTER messages in its registration database. If the REGISTER message is received from a User-type IP Group, the device sends the message to the configured destination.

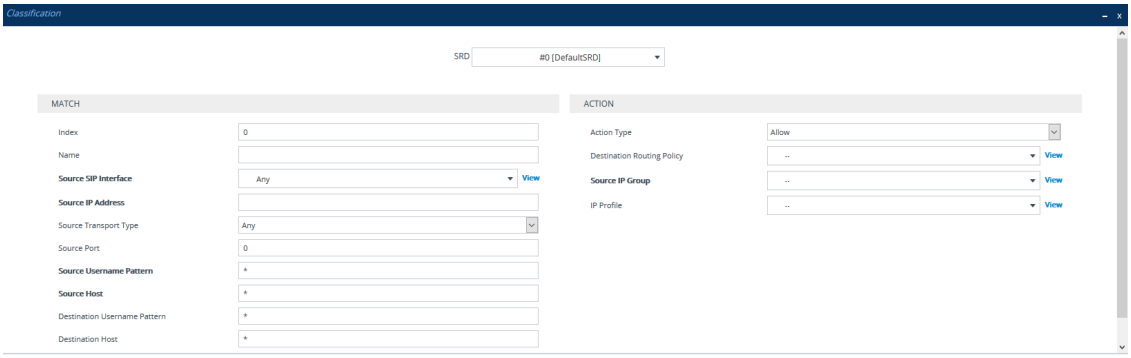
The flowchart below illustrates the classification process:



The following procedure describes how to configure Classification rules through the Web interface. You can also configure it through ini file [Classification] or CLI (`configure voip > sbc classification`).

➢ To configure a Classification rule:

1. Open the Classification table (**Setup** menu > **Signaling & Media** tab > **SBC** folder > **Classification Table**).
2. Click **New**; the following dialog box appears:



3. Configure the Classification rule according to the parameters described in the table below.
4. Click **Apply**.

Table 25-1: Classification Table Parameter Descriptions

Parameter	Description
'SRD' `srd-name` [Classification_SRDName]	Assigns an SRD to the rule as a matching characteristic for the incoming SIP dialog. If only one SRD is configured in the SRDs table, the SRD is assigned to the rule by default. If multiple SRDs are configured in the SRDs table, no value is assigned. To configure SRDs, see Configuring SRDs. **Note:** The parameter is mandatory.
Match	
'Index' [Classification_Index]	Defines an index number for the new table row. **Note:** Each row must be configured with a unique index.
'Name' `classification-name` [Classification_ ClassificationName]	Defines a descriptive name, which is used when associating the row in other tables. The valid value is a string of up to 40 characters. By default, no name is defined. **Note:** Each row must be configured with a unique name.
'Source SIP Interface' `src-sip-interface-name` [Classification_ SrcSIPInterfaceName]	Assigns a SIP Interface to the rule as a matching characteristic for the incoming SIP dialog. The default is **Any** (i.e., all SIP Interfaces belonging to the SRD assigned to the rule). **Note:** The SIP Interface must belong to the SRD assigned to the rule (see the 'SRD' parameter in the table).
'Source IP Address' src-ip-address [Classification_SrcAddress]	Defines a source IP address as a matching characteristic for the incoming SIP dialog. The valid value is an IP address in dotted-decimal notation.

Parameter	Description
	In addition, the following wildcards can be used: ■ "x" wildcard: represents single digits. For example, 10.8.8.xx represents all addresses between 10.8.8.10 and 10.8.8.99. ■ Asterisk (*) wildcard: represents any number between 0 and 255. For example, 10.8.8.* represents all addresses between 10.8.8.0 and 10.8.8.255. By default, no value is defined (i.e., any source IP address is accepted). **Note:** ■ The parameter is applicable only to Server-type IP Groups. ■ If the IP address is unknown (i.e., configured for the associated Proxy Set as an FQDN), it is recommended to classify incoming dialogs based on Proxy Set (instead of using a Classification rule). For more information on classification by Proxy Set or by Classification rule, see the note bulletin in the beginning of this section.
'Source Transport Type' `src-transport-type` [Classification_ SrcTransportType]	Defines the source transport type as a matching characteristic for the incoming SIP dialog. ■ [-1] **Any** = (Default) All transport types ■ [0] **UDP** ■ [1] **TCP** ■ [2] **TLS** ■ [3] **SCTP**
'Source Port' `src-port` [Classification_SrcPort]	Defines the source port number as a matching characteristic for the incoming SIP dialog. By default, no value is defined.
'Source Username Pattern' `src-user-name-pattern` [Classification_ SrcUsernamePrefix]	Defines the source URI user part as a matching characteristic for the incoming SIP dialog. The URI is typically located in the SIP From header. However, you can configure the SIP header from where the device obtains the source URI, in the IP Groups table ('Source URI Input' parameter). For more information on how the device obtains the URI, see SIP Dialog Initiation Process.

Parameter	Description
	You can use special patterns (notations) to denote the user part. For example, if you want to match this rule to user parts whose last four digits (i.e., suffix) are 4 followed by any three digits (e.g., 4008), then configure this parameter to "(4xxx)", without the quotation marks. For available patterns, see Dialing Plan Notation for Routing and Manipulation. The valid value is a string of up to 60 characters. The default is the asterisk (*) symbol, meaning any source user part. **Note:** For REGISTER requests, the source URI is obtained from the To header.
'Source Host' `src-host` [Classification_SrcHost]	Defines the prefix of the source URI host name as a matching characteristic for the incoming SIP dialog. The URI is typically located in the SIP From header. However, you can configure the SIP header from where the device obtains the source URI, in the IP Groups table ('Source URI Input' parameter). For more information on how the device obtains this URI, see Call Processing of SIP Dialog Requests. The default is the asterisk (*) symbol, which represents any source host prefix. **Note:** For REGISTER requests, the source URI is obtained from the To header.
'Destination Username Pattern' `dst-user-name-pattern` [Classification_ DestUsernamePrefix]	Defines the destination Request-URI user part as a matching characteristic for the incoming SIP dialog. You can use special patterns (notations) to denote the user part. For example, if you want to match this rule to user parts whose last four digits (i.e., suffix) are 4 followed by any three digits (e.g., 4008), then configure this parameter to "(4xxx)", without the quotation marks. For available patterns, see Dialing Plan Notation for Routing and Manipulation. The valid value is a string of up to 60 characters. The default is the asterisk (*) symbol, meaning any destination user part.
'Destination Host' `dst-host` [Classification_DestHost]	Defines the prefix of the destination Request-URI host name as a matching characteristic for the incoming SIP dialog.

Parameter	Description
	The default is the asterisk (*) symbol, which represents any destination host prefix.
'Message Condition' `message-condition-name` [Classification_ MessageConditionName]	Assigns a Message Condition rule to the Classification rule as a matching characteristic for the incoming SIP dialog. By default, no value is defined. To configure Message Condition rules, see Configuring Message Condition Rules.
Action	
'Action Type' `action-type` [Classification_ActionType]	Defines a whitelist or blacklist for the matched incoming SIP dialog. ■ [0] **Deny** = Blocks incoming SIP dialogs that match the characteristics of the rule (blacklist). ■ [1] **Allow** = (Default) Allows incoming SIP dialogs that match the characteristics of the rule (whitelist) and assigns it to the associated IP Group.
'Destination Routing Policy' `dest-routing-policy` [Classification_ DestRoutingPolicy]	Assigns a Routing Policy to the matched incoming SIP dialog. The assigned Routing Policy overrides the Routing Policy assigned to the SRD (in the SRDs table). The option to assign Routing Policies to Classification rules is useful in deployments requiring different routing and manipulation rules for specific calls pertaining to the **same** SRD. In such scenarios, you need to configure multiple Classification rules for the same SRD, where for some rules no Routing Policy is assigned (i.e., the SRD's assigned Routing Policy is used) while for others a different Routing Policy is specified to override the SRD's assigned Routing Policy. By default, no value is defined. To configure Routing Policies, see Configuring SBC Routing Policy Rules.
'IP Group Selection' `ip-group-selection` [Classification_ IPGroupSelection]	Defines how the incoming SIP dialog is classified to an IP Group. ■ [0] **Source IP Group** = (Default) The SIP dialog is classified to the IP Group that is specified in the 'Source IP Group' parameter (see below). ■ [1] **Tagged IP Group** = The SIP dialog is classified to an

Parameter	Description
	IP Group based on source tag, which is specified in the 'IP Group Tag Name' parameter (see below). For more information on Classification of incoming SIP dialogs to IP Groups using tags, see Configuring Classification Based on Tags on page 714.
'Source IP Group' src-ip-group-name [Classification_ SrcIPGroupName]	Assigns an IP Group to the matched incoming SIP dialog. By default, no value is defined. To configure IP Groups, see Configuring IP Groups. **Note:** ■ The parameter is applicable only if you configure the 'IP Group Selection' parameter to **Source IP Group**. ■ The IP Group must be associated with the assigned SRD (see the 'SRD' parameter in the table).
'IP Group Tag Name' ip-group-tag-name [Classification_ IpGroupTagName]	Defines the source tag of the incoming SIP dialog. The tag is used for classifying the SIP dialog to an IP Group. The tag is obtained from the Call Setup Rule that is associated with the SIP Interface on which the dialog is received. The valid value is a string of up to 70 characters. The default value is "default" (without quotation marks), which must be used when the resultant tag from the Call Setup Rule is only a value (e.g., "Ireland"). If the resultant tag is a name=value (e.g., "Country=Ireland"), then configure the parameter with the **name** only (e.g., "Country"). Only **one** tag name can be configured. For more information on Classification of incoming SIP dialogs to IP Groups using tags, see Configuring Classification Based on Tags on page 714. **Note:** The parameter is applicable only if you configure the 'IP Group Selection' parameter to **Tagged IP Group**.
'IP Profile' ip-profile-id [Classification_ IpProfileName]	Assigns an IP Profile to the matched incoming SIP dialog. The assigned IP Profile overrides the IP Profile assigned to the IP Group (in the IP Groups table) to which the SIP dialog is classified. Therefore, assigning an IP Profile during classification allows you to assign different IP Profiles to specific users (calls) that belong to the same IP Group (User or Server type). For example, you can configure two Classification rules to classify incoming calls to the same IP Group. However, one

Parameter	Description
	Classification rule is a regular rule that doesn't specify any IP Profile (IP Profile assigned to IP Group is used), while the second rule is configured with an additional matching characteristic for the source hostname prefix (e.g., "abcd.com") and with an additional action that assigns a different IP Profile. By default, no value is defined. **Note:** For User-type IP Groups, if a user is already registered with the device (from a previous, initial classification process), the device classifies subsequent INVITE requests from the user according to the device's users database instead of the Classification table. In such a scenario, the same IP Profile that was previously assigned to the user by the Classification table is also used (in other words, the device's users database stores the associated IP Profile).

Classification Based on URI of Selected Header Example

The following example describes how to configure classification of incoming calls to IP Groups, based on source URI in a specific SIP header. The example assumes the following incoming INVITE message:

```
INVITE sip:8000@10.33.4.226 SIP/2.0
Via: SIP/2.0/UDP 10.33.4.226;branch=z9hG4bKVEBTDAHSUYRTEXEDEGJY
From: <sip:100@10.33.4.226>;tag=YSQQKXXREVDPYPTNFMWG
To: <sip:8000@10.33.4.226>
Call-ID: FKPNOYRNKROIMEGBSSKS@10.33.4.226
CSeq: 1 INVITE
Contact: <sip:100@10.33.4.226>
Route: <sip:2000@10.10.10.10.10>,<sip:300@10.10.10.30>
Supported: em,100rel,timer,replaces
P-Called-Party-ID: <sip:1111@10.33.38.1>
User-Agent: Sip Message Generator V1.0.0.5
Content-Length: 0
```

1. In the Classification table, add the following classification rules:

Index	Source Username Pattern	Destination Username Pattern	Destination Host	Source IP Group
0	333	-	-	1

Index	Source Username Pattern	Destination Username Pattern	Destination Host	Source IP Group
1	1111	2000	10.10.10.10	2

2. In the IP Groups table, add the following IP Groups:

Index	Source URI Input	Destination URI Input
1	-	-
2	P-Called-Party-ID	Route

In the example, a match exists only for Classification Rule #1. This is because the source (1111) and destination (2000) username prefixes match those in the INVITE's P-Called-Party-ID header (i..e., "<sip:1111@10.33.38.1>") and Route header (i.e., "<sip:2000@10.10.10.10.10>"), respectively. These SIP headers were determined in IP Group 2.

Configuring Classification Based on Tags

You can classify incoming SIP dialogs to IP Groups, using tags (source tags) that are obtained from Call Setup Rules associated with the SIP Interfaces on which dialogs are received. Using tags can significantly reduce the number of required Classification rules. In some scenarios, a single Classification rule may suffice.

Classification based on tags includes the following stages:

1. The device determines the tag of the incoming SIP dialog by running a Call Setup Rule that is associated with the SIP Interface on which the dialog is received. The Call Setup Rule on SIP Interfaces can be based **only** on synchronous queries. You can configure the Call Setup Rule to generate a tag with a name and value (e.g., "Country=Ireland") or only a value (e.g., "Ireland").
2. The device searches the Classification table for a matching rule based on the SIP Interface (and optionally, any other existing matching properties) as well as the tag. The tag can be a name (e.g., "Country"), or "default" if the tag only has a value (e.g., "Ireland").
3. The device searches the IP Groups table for an IP Group that is configured with the tag from the Call Setup Rule (name=value or value only) and if found, classifies the dialog to that IP Group.

- Classification based on tags is done only if classification based on user registration and Proxy Set fail.
- The IP Group Set table is not used for classification (i.e., ignores tags).

The following procedure describes how to configure incoming SIP dialog classification based on tags. The procedure is based on an example that uses Dial Plan tags to classify calls to three different IP Groups:

- Calls with source number (user) 410 are classified to IP Group-1
- Calls with source number (user) 420 are classified to IP Group-2
- Calls with source number (user) 430 are classified to IP Group-3

➢ **To configure Classification based on tags:**

1. Open the Dial Plan table (see Configuring Dial Plans on page 609), and then configure Dial Plan tags. In our example, the following Dial Plan rules are configured for Dial Plan "ITSP":

Name	Prefix	Tag
Rule1	410	Country=Ireland
Rule2	420	Country=Scotland
Rule3	430	Country=England

2. Open the Call Setup Rules table (see Configuring Call Setup Rules on page 595), and then configure Call Setup Rules for obtaining source tags of incoming SIP dialogs. In our example, the following Call Setup Rule is configured:

General	
'Rules Set ID'	**1**
'Request Type'	**Dial Plan**
'Request Target'	**ITSP**
'Request Key'	**Param.Call.Src.User**
'Condition '	**DialPlan.Found exists**
Action	
'Action Subject'	**SrcTags**
'Action Type'	**Modify**
'Action Value'	**DialPlan.Result**

3. Open the SIP Interfaces table (see Configuring SIP Interfaces on page 434), and then configure a SIP Interface with the 'Call Setup Rules Set ID' parameter set to the 'Rules Set ID' value of your Call Setup Rules. In our example, the SIP Interface is named "SIPIfx-Tags" and the parameter is configured to 1.
4. Open the Classification table, and then configure a rule with the following:

Match	
'Source SIP Interface'	SIPIfx-Tags (or select **Any**)
Action	
'IP Group Selection'	**Tagged IP Group**
'IP Group Tag Name'	**Country** **Note:** Enter the tag's name only. If the tag only has a value, then enter "default" (without quotation marks).

5. Open the IP Groups table (see Configuring IP Groups on page 451), and then configure IP Groups with the 'Tags' parameter set to the appropriate tag. If the source tag has a name and value, then configure the parameter as name=value (e.g., "Country=Ireland"). If it only has a value, then configure it with the value. In our example, the following IP Groups are configured:

Name	Tags
IPGroup-1	Country=Ireland
IPGroup-2	Country=Scotland
IPGroup-3	Country=England

Configuring SBC IP-to-IP Routing

The IP-to-IP Routing table lets you configure up to 3,750 SBC IP-to-IP routing rules.

Configuration of IP-to-IP routing rules includes two areas:

- **Match:** Defines the characteristics of the incoming SIP dialog message (e.g., IP Group from which the message is received).
- **Action:** Defines the action that is done if the incoming call matches the characteristics of the rule (i.e., routes the call to the specified destination).

The device searches the table from **top to bottom** for the **first** rule that matches the characteristics of the incoming call. If it finds a matching rule, it sends the call to the destination configured for that rule. If it doesn't find a matching rule, it rejects the call.

Configure stricter rules higher up in the table than less strict rules to ensure the desired rule is used to route the call. *Strict* refers to the number of matching characteristics configured for the rule. For example, a rule configured with source host name and source IP Group as matching characteristics is stricter than a rule configured with only source host name. If the rule configured with only source host name appears higher up in the table, the device ("erroneously") uses the rule to route calls matching this source host name (even if they also match the rule appearing lower down in the table configured with the source IP Group as well).

The IP-to-IP Routing table lets you route incoming SIP dialog messages (e.g., INVITE) to any of the following IP destinations:

- According to registered user Contact listed in the device's registration database (only for User-type IP Groups).
- IP Group - the destination is the address configured for the Proxy Set associated with the IP Group.
- IP Group Set - the destination can be based on multiple IP Groups for load balancing, where each call may be sent to a different IP Group within the IP Group Set depending on the IP Group Set's definition.
- Routing tag - the device sends the call to an IP Group (or IP Group Set) based on a destination Dial Plan tag that corresponds to the destination (called) prefix number.
- IP address in dotted-decimal notation or FQDN. Routing to a host name can be resolved using NAPTR/SRV/A-Record.
- Request-URI of incoming SIP dialog-initiating requests.
- Any registered user in the registration database. If the Request-URI of the incoming INVITE exists in the database, the call is sent to the corresponding contact address specified in the database.
- According to result of an ENUM query.
- Hunt Group - used for call survivability of call centers (see Configuring Call Survivability for Call Centers).
- According to result of LDAP query (for more information on LDAP-based routing, see Routing Based on LDAP Active Directory Queries).
- Third-party routing server, which determines the destination (next hop) of the call (IP Group). The IP Group represents the next device in the routing path to the final destination. For more information, see Centralized Third-Party Routing Server.
- Back to the sender of the incoming message, where the reply can be a SIP response code or a 3xx redirection response (with an optional Contact field to where the sender must re-send the message).

The following figure summarizes the destination types:

To configure and apply an IP-to-IP Routing rule, the rule must be associated with a Routing Policy. The Routing Policy associates the routing rule with an SRD(s). Therefore, the Routing Policy lets you configure routing rules for calls belonging to specific SRD(s). However, as multiple Routing Policies are relevant only for multi-tenant deployments (if needed), for most deployments, only a single Routing Policy is required. As the device provides a default Routing Policy ("Default_SBCRoutingPolicy"), when only one Routing Policy is required, the device automatically assigns the default Routing Policy to the routing rule. If you are implementing LDAP-based routing (with or without Call Setup Rules) and/or Least Cost Routing (LCR), you need to configure these settings for the Routing Policy (regardless of the number of Routing Policies employed). For more information on Routing Policies, see Configuring SBC Routing Policy Rules.

The IP-to-IP Routing table also provides the following features:

- **Alternative Routing:** In addition to the alternative routing/load balancing provided by the Proxy Set associated with the destination IP Group, the table allows the configuration of alternative routes where if a route fails, the next adjacent (below) rule in the table that is configured to **Alt Route Ignore/Consider Inputs** are used. The alternative routing rules can be set to enforce the input matching criteria or to ignore any matching criteria. Alternative routing occurs upon one of the following conditions:
  - A request sent by the device is responded with one of the following:

- SIP response code (e.g., 4xx, 5xx, and 6xx) that is also configured for an Alternative Reasons Set (see Configuring SIP Response Codes for Alternative Routing Reasons) assigned to the IP Group ('SBC Alternative Routing Reasons Set' parameter).
- SIP 408 Timeout or no response (after timeout).

- The DNS resolution includes IP addresses that the device has yet to try (for the current call).

Messages are re-routed with the same SIP Call-ID and CSeq header fields (increased by 1).

If the Proxy Set (see Configuring Proxy Sets) associated with the destination of the call is configured with multiple IP addresses, the device first attempts to route the call to one of these IP addresses, starting with the first listed address. Only when the call cannot be routed to any of the Proxy Set's IP addresses does the device search the IP-to-IP Routing table for an alternative routing rule for the call.

- **Load Balancing:** You can implement load balancing of calls, belonging to the same source, between a set of destination IP Groups known as an *IP Group Set*. The IP Group Set can include up to five IP Groups (Server-type and/or Gateway-type only) and the chosen IP Group depends on the configured load-balancing policy (e.g., Round Robin). To configure the feature, you need to first configure an IP Group Set (see Configuring IP Group Sets), and then assign it to a routing rule with 'Destination Type' configured to **IP Group Set**.
- **Re-routing SIP Requests:** This table enables you to configure "re-routing" rules of requests (e.g., INVITEs) that the device sends upon receipt of SIP 3xx responses or REFER messages. These rules are configured for destinations that do not support receipt of 3xx or REFER and where the device handles the requests locally (instead of forwarding the 3xx or REFER to the destination).
- **Least Cost Routing (LCR):** If the LCR feature is enabled, the device searches the routing table for matching routing rules and then selects the one with the lowest call cost. The call cost of the routing rule is done by assigning it a Cost Group. To configure Cost Groups, see Least Cost Routing. If two routing rules have identical costs, then the rule appearing higher up in the table (i.e., first-matched rule) is used. If a selected route is unavailable, the device uses the next least-cost routing rule. However, even if a matched rule is not assigned a Cost Group, the device can select it as the preferred route over other matched routing rules that are assigned Cost Groups, according to the default LCR settings configured for the assigned Routing Policy (see Configuring SBC Routing Policy Rules).
- **Call Forking:** The IP-to-IP Routing table can be configured to route an incoming IP call to multiple destinations (call forking). The incoming call can be routed to multiple destinations of any type such as an IP Group or IP address. The device forks the call by sending simultaneous INVITE messages to all the specified destinations. It handles the multiple SIP dialogs until one of the calls is answered and then terminates the other SIP dialogs.

  Call forking is configured by creating a Forking group. A Forking group consists of a main routing rule ('Alternative Route Options' set to **Route Row**) whose 'Group Policy' is set to **Forking**, and one or more associated routing rules ('Alternative Route Options' set to **Group**

Member Ignore Inputs or **Group Member Consider Inputs**). The group members must be configured in contiguous table rows to the main routing rule. If an incoming call matches the input characteristics of the main routing rule, the device routes the call to its destination and all those of the group members.

An alternative routing rule can also be configured for the Forking group. The alternative route is used if the call fails for the Forking group (i.e., main route and all its group members). The alternative routing rule must be configured in the table row immediately below the last member of the Forking group. The 'Alternative Route Options' of this alternative route must be set to **Alt Route Ignore Inputs** or **Alt Route Consider Inputs**. The alternative route can also be configured with its own forking group members, where if the device uses the alternative route, the call is also sent to its group members. In this case, instead of setting the alternative route's 'Group Policy' to **None**, you must set it to **Forking**. The group members of the alternative route must be configured in the rows immediately below it.

The LCR feature can also be employed with call forking. The device calculates a maximum call cost for each Forking group and routes the call to the Forking group with the lowest cost. Thus, even if the call can successfully be routed to the main routing rule, a different routing rule can be chosen (even an alternative route, if configured) based on LCR. If routing to one Forking group fails, the device tries to route the call to the Forking group with the next lowest cost (main or alternative route), and so on. The prerequisite for this functionality is that the incoming call must successfully match the input characteristics of the main routing rule.

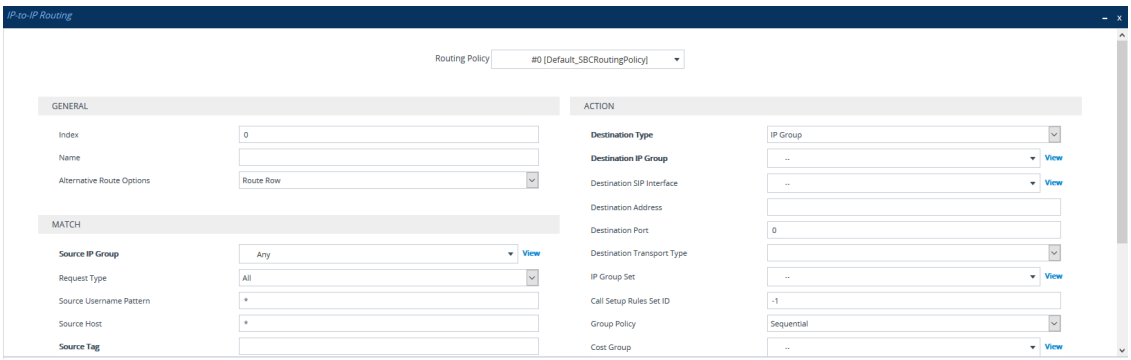
- **Dial Plan Tags Representing Source / Destination Numbers:** If your deployment includes calls of many different called (source URI user name) and/or calling (destination URI user name) numbers that need to be routed to the same destination, you can employ user-defined tags to represent these numbers. Thus, instead of configuring many routing rules, you can configure only one routing rule using the tag as the source and destination number matching characteristics, and a destination for the calls. For more information, see Using Dial Plan Tags for Matching Routing Rules.
- **Dial Plan Tags for Determining Destination IP Group:** Instead of configuring multiple routing rules, you can configure a single routing rule with a specific "destination" Dial Plan tag. The device uses the tag to determine the destination IP Group. For more information, see Using Dial Plan Tags for Routing Destinations.
- **Fax Rerouting:** You can configure the device to reroute incoming calls that it identifies as fax calls to a new IP destination. For more information, see Configuring Rerouting of Calls to Fax Destinations.

Call forking is not applicable to LDAP-based routing.

The following procedure describes how to configure IP-to-IP routing rules through the Web interface. You can also configure it through ini file [IP2IPRouting] or CLI (`configure voip > sbc routing ip2ip-routing`).

➢ To configure an IP-to-IP routing rule:

1. Open the IP-to-IP Routing table (**Setup** menu > **Signaling & Media** tab > **SBC** folder > **Routing** > **IP-to-IP Routing**).
2. Click **New**; the following dialog box appears:



3. Configure an IP-to-IP routing rule according to the parameters described in the table below.
4. Click **Apply**.

Table 25-2: IP-to-IP Routing Table Parameter Descriptions

Parameter	Description
'Routing Policy' `sbc-routing-policy-name` [IP2IPRouting_RoutingPolicyName]	Assigns a Routing Policy to the rule. The Routing Policy associates the rule with an SRD(s). The Routing Policy also defines default LCR settings as well as the LDAP servers used if the routing rule is based on LDAP routing (and Call Setup Rules). If only one Routing Policy is configured in the Routing Policies table, the Routing Policy is automatically assigned. If multiple Routing Policies are configured, no value is assigned. To configure Routing Policies, see Configuring SBC Routing Policy Rules. **Note:** The parameter is mandatory.
General	
'Index' [IP2IPRouting_Index]	Defines an index number for the new table row. **Note:** Each row must be configured with a unique index.
'Name' `route-name` [IP2IPRouting_RouteName]	Defines a descriptive name, which is used when associating the row in other tables. The valid value is a string of up to 40 characters. By default, no value is defined.
'Alternative Route Options'	Determines whether this routing rule is the main routing rule or an alternative routing rule (to the rule defined directly above it in the

Parameter	Description
`alt-route-options` [IP2IPRouting_AltRouteOptions]	table). ■ [0] **Route Row** = (Default) Main routing rule - the device first attempts to route the call to this route if the incoming SIP dialog's input characteristics matches this rule. ■ [1] **Alternative Route Ignore Inputs** = If the call cannot be routed to the main route (Route Row), the call is routed to this alternative route regardless of the incoming SIP dialog's input characteristics. ■ [2] **Alternative Route Consider Inputs** = If the call cannot be routed to the main route (Route Row), the call is routed to this alternative route only if the incoming SIP dialog matches this routing rule's input characteristics. ■ [3] **Group Member Ignore Inputs** = This routing rule is a member of the Forking routing rule. The incoming call is also forked to the destination of this routing rule. The matching input characteristics of the routing rule are ignored. ■ [4] **Group Member Consider Inputs** = This routing rule is a member of the Forking routing rule. The incoming call is also forked to the destination of this routing rule only if the incoming call matches this rule's input characteristics. **Note:** ■ The alternative routing entry ([1] or [2]) must be defined in the next consecutive table entry index to the Route Row entry (i.e., directly below it). For example, if Index 4 is configured as a Route Row, Index 5 must be configured as the alternative route. ■ The Forking Group members must be configured in a table row that is immediately below the main Forking routing rule, or below an alternative routing rule for the main rule, if configured. ■ For IP-to-IP alternative routing, configure alternative routing based on the receipt of specific SIP SIP responses (see Configuring SIP Response Codes for Alternative Routing Reasons). However, if no response, ICMP, or a SIP 408 response is received, the device attempts to use the alternative route even if you haven't configured any SIP responses for alternative routing. ■ Multiple alternative route entries can be configured (e.g., Index

Parameter	Description
	1 is the main route - Route Row - and indices 2 through 4 are configured as alternative routes).
Match	
'Source IP Group' `src-ip-group-name` [IP2IPRouting_SrcIPGroupName]	Defines the IP Group from where the IP call is received (i.e., the IP Group that sent the SIP dialog). Typically, the IP Group of an incoming SIP dialog is determined (or classified) using the Classification table (see Configuring Classification Rules). The default is **Any** (i.e., any IP Group). **Note:** The selectable IP Group for the parameter depends on the assigned Routing Policy (in the 'Routing Policy' parameter in this table). For more information, see Configuring SBC Routing Policy Rules.
'Request Type' `request-type` [IP2IPRouting_RequestType]	Defines the SIP dialog request type (SIP Method) of the incoming SIP dialog. ■ [0] **All** (default) ■ [1] **INVITE** ■ [2] **REGISTER** ■ [3] **SUBSCRIBE** ■ [4] **INVITE and REGISTER** ■ [5] **INVITE and SUBSCRIBE** ■ [6] **OPTIONS** **Note:** ■ For User-type IP Groups, if you also need to send REGISTER messages received from this IP Group, then it is highly recommended that the configured destination of the routing rule is a Server-type IP Group and **not** an IP address (configured by the 'Destination Type' parameter). If you need to send non-REGISTER messages (e.g., INVITE) to a destination that is configured as an IP address, then you need to configure two IP-to-IP Routing rules for this User-type IP Group, one for routing REGISTER messages and one for routing non-REGISTER messages. ■ If the device receives a REFER message, it searches again for a matching routing rule in the IP-to-IP Routing table and then forwards the message to the destination configured of the matched rule.

Parameter	Description
'Source Username Pattern' src-user-name-pattern [IP2IPRouting_SrcUsernamePrefix]	Defines the user part of the incoming SIP dialog's source URI (usually the From URI). You can use special patterns (notations) to denote the user part. For example, if you want to match this rule to user parts whose last four digits (i.e., suffix) are 4 followed by any three digits (e.g., 4008), then configure this parameter to "(4xxx)". To denote calls without a user part in the URI, use the dollar ($) sign. For available patterns, see Dialing Plan Notation for Routing and Manipulation. The valid value is a string of up to 60 characters. The default is the asterisk (*) symbol (i.e., any user part). If this rule is not required, leave this field empty. **Note:** If you need to route calls of many different source URI user names to the same destination, you can use tags (see 'Source Tags' parameter below) instead of this parameter.
'Source Host' src-host [IP2IPRouting_SrcHost]	Defines the host part of the incoming SIP dialog's source URI (usually the From URI). The default is the asterisk (*) symbol (i.e., any host name). If this rule is not required, leave this field empty.
'Source Tags' src-tags [IP2IPRouting_SrcTags]	Assigns a tag to denote source URI user names corresponding to the tag configured in the associated Dial Plan. The valid value is a string of up to 70 characters. By default, no value is defined. You can configure the parameter with up to five tags, where each tag is separated by a semicolon (;). However, you can configure only up to four tags containing a name and value (e.g., Country=Ireland), and **one** tag containing a value only (e.g., Ireland). If you are configuring multiple tags in the name=value format, the names of each tag must be unique (e.g., **Country**=Ireland;**Land**=Scotland). The following example configures the maximum number of tags (i.e., four name=value tags and one value-only tag): Country=Ireland;Country2=Scotland;Country3=RSA;Country4=Canada;USA. To configure tags, see Configuring Dial Plans. **Note:** ■ The tag is case insensitive. ■ Make sure that you assign the Dial Plan in which you have configured the tag, to the related IP Group or SRD. ■ Instead of using tags and configuring the parameter, you can

Parameter	Description
	use the 'Source Username Pattern' parameter to specify a specific URI source user or all source users.
'Destination Username Pattern' dst-user-name-pattern [IP2IPRouting_ DestUsernamePrefix]	Defines the incoming SIP dialog's destination URI (usually the Request URI) user part. You can use special patterns (notations) to denote the user part. For example, if you want to match this rule to user parts whose last four digits (i.e., suffix) are 4 followed by any three digits (e.g., 4008), then configure this parameter to "(4xxx)". To denote calls without a user part in the URI, use the dollar ($) sign. For available patterns, see Dialing Plan Notation for Routing and Manipulation. The valid value is a string of up to 60 characters. The default is the asterisk (*) symbol (i.e., any user part). If this rule is not required, leave this field empty. **Note:** If you need to route calls of many different destination URI user names to the same destination, you can use tags (see 'Source Tags' parameter below) instead of this parameter.
'Destination Host' dst-host [IP2IPRouting_ DestHost]	Defines the host part of the incoming SIP dialog's destination URI (usually the Request-URI). The default is the asterisk (*) symbol (i.e., any destination host). If this rule is not required, leave this field empty.
'Destination Tags' dest-tags [IP2IPRouting_ DestTags]	Assigns a prefix tag to denote destination URI user names corresponding to the tag configured in the associated Dial Plan. The valid value is a string of up to 70 characters. By default, no value is defined. You can configure the parameter with up to five tags, where each tag is separated by a semicolon (;). However, you can configure only up to four tags containing a name and value (e.g., Country=Ireland), and **one** tag containing a value only (e.g., Ireland). If you are configuring multiple tags in the name=value format, the names of each tag must be unique (e.g., **Country**=Ireland;**Land**=Scotland). The following example configures the maximum number of tags (i.e., four name=value tags and one value-only tag): Country=Ireland;Country2=Scotland;Country3=RSA;Country4=Canada;USA. To configure prefix tags, see Configuring Dial Plans. **Note:** ■ The tag is case insensitive. ■ Make sure that you assign the Dial Plan in which you have

Parameter	Description
	configured the prefix tag, to the related IP Group or SRD. ■ Instead of using tags and configuring the parameter, you can use the 'Destination Username Pattern' parameter to specify a specific URI destination user or all destinations users.
'Message Condition' `message-condition-name` [IP2IPRouting_MessageConditionName]	Assigns a SIP Message Condition rule to the IP-to-IP Routing rule. To configure Message Condition rules, see Configuring Message Condition Rules.
'Call Trigger' `trigger` [IP2IPRouting_Trigger]	Defines the reason (i.e., trigger) for re-routing (i.e., alternative routing) the SIP request. ■ [0] **Any** = (Default) This routing rule is used for all scenarios (re-routes and non-re-routes). ■ [1] **3xx** = Re-routes the request if it was triggered as a result of a SIP 3xx response. ■ [2] **REFER** = Re-routes the INVITE if it was triggered as a result of a REFER request. ■ [3] **3xx or REFER** = Applies to options [1] and [2]. ■ [4] **Initial only** = This routing rule is used for regular requests that the device forwards to the destination. This rule is not used for re-routing of requests triggered by the receipt of REFER or 3xx. ■ [5] **Broken Connection** = If the device detects a broken RTP connection during the call and the Broken RTP Connection feature is enabled (IpProfile_DisconnectOnBrokenConnection parameter is configured to [2]), you can use this option as an explicit matching characteristics to route the call to an alternative destination. Therefore, for alternative routing upon broken RTP detection, position the routing rule configured with this option above the regular routing rule associated with the call. Such a configuration setup ensures that the device uses this alternative routing rule only when RTP broken connection is detected. ■ [6] **Fax Rerouting** = Reroutes the INVITE to a fax destination (different IP Group) if it is identified as a fax call. For more

Parameter	Description
	information, see Configuring Rerouting of Calls to Fax Destinations.
'ReRoute IP Group' re-route-ip-group-id [IP2IPRouting_ ReRouteIPGroupName]	Defines the IP Group that initiated (sent) the SIP redirect response (e.g., 3xx) or REFER message. This parameter is typically used for rerouting requests (e.g., INVITEs) when interworking is required for SIP 3xx redirect responses or REFER messages. For more information, see Interworking SIP 3xx Redirect Responses and Interworking SIP REFER Messages, respectively. The parameter functions together with the 'Call Trigger' parameter (in the table). The default is **Any** (i.e., any IP Group). **Note:** The selectable IP Group for the parameter depends on the assigned Routing Policy (in the 'Routing Policy' parameter in this table). For more information, see Configuring SBC Routing Policy Rules.
Action	
'Destination Type' dst-type [IP2IPRouting_ DestType]	Determines the destination type to which the outgoing SIP dialog is sent. ■ [0] **IP Group** = (Default) The SIP dialog is sent to the IP Group as defined in the 'Destination IP Group' (IP2IPRouting_ DestIPGroupName) parameter. For more information on the actual address, see the 'Destination IP Group' parameter. ■ [1] **Dest Address** = The SIP dialog is sent to the address configured in the following parameters: 'Destination Address', 'Destination Port' and 'Destination Transport Type'. ■ [2] **Request URI** = The SIP dialog is sent to the address indicated in the incoming Request-URI. If the parameters 'Destination Port' and 'Destination Transport Type' are configured, the incoming Request-URI parameters are overridden and these parameters take precedence. ■ [3] **ENUM** = An ENUM query is sent to include the destination address. If the parameters 'Destination Port' and 'Destination Transport Type' are configured, the incoming Request-URI parameters are overridden and these parameters take precedence. ■ [4] **Hunt Group** = Used for call center survivability. For more information, see Configuring Call Survivability for Call Centers. ■ [5] **Dial Plan** = (**For Backward Compatibility Only - see Note**

Parameter	Description
	below) The IP destination is determined by a Dial Plan index of the loaded Dial Plan file. The syntax of the Dial Plan index in the Dial Plan file is as follows: <destination / called prefix number>,0,<IP destination> Note that the second parameter "0" is ignored. An example of a configured Dial Plan (# 6) in the Dial Plan file is shown below: [PLAN6] 200,0,10.33.8.52 ; called prefix 200 is routed to destination 10.33.8.52 201,0,10.33.8.52 300,0,itsp.com ; called prefix 300 is routed to destination itsp.com Once the Dial Plan is defined, you need to assign it (0 to 7) to the routing rule as the destination in the 'Destination Address' parameter, where "0" denotes [PLAN1], "1" denotes [PLAN2], and so on. ■ [7] **LDAP** = LDAP-based routing. Make sure that the Routing Policy assigned to the routing rule is configured with the LDAP Server Group for defining the LDAP server(s) to query. ■ [9] **Routing Server** = Device sends a request to a third-party routing server for an appropriate destination (next hop) for the matching call. ■ [10] **All Users** = The device checks if the SIP Request-URI (i.e., destination user) in the incoming INVITE is registered in its' users database. If registered, the device sends the INVITE to the address of the corresponding contact that is specified in the database. If the Request-URI is not registered, the call is rejected. If the incoming SIP dialog is a REGISTER message, the device acts as a registrar and only responds to the sender of the request (200 OK) without sending the REGISTER message to a destination (i.e., termination of REGISTER messages). ■ [11] **IP Group Set** = The device employs load balancing and routes the call to one of the IP Groups in the IP Group Set assigned using the 'IP Group Set' parameter (below). ■ [12] **Destination Tag** = The device routes the call to an IP Group determined by Dial Plan tags. The tag is specified in the 'Routing Tag Name' parameter (below). For more information

Parameter	Description
	on using tags to determine destination IP Group, see Using Dial Plan Tags for Routing Destinations. ■ [13] **Internal** = Instead of sending the incoming SIP dialog to another destination, the device replies to the sender of the dialog with a SIP response code or a redirection response, configured by the 'Internal Action' (IP2IPRouting_ InternalAction) parameter in this table (see below). **Note:** ■ Use the **Dial Plan** option only for backward compatibility purposes; otherwise, use prefix tags as described in Configuring Dial Plans. ■ If you configure the parameter to **Dest Address**, **Request URI**, **ENUM**, **Dial Plan** or **LDAP**, you must specify a destination IP Group using the 'Destination IP Group' parameter, even though these calls are **not** sent to the specified IP Group (i.e., its associated Proxy Set). This allows you to associate other configuration entities (such as an IP Profile) that are assigned to the IP Group, with the destination of these calls. If you do not specify a destination IP Group, the device uses its own logic in choosing a destination IP Group (and thus its associated configuration entities) for the routing rule. ■ You can configure up to 20 IP-to-IP Routing rules whose 'Destination Type' is **Internal**.
'Destination IP Group' `dst-ip-group-name` [IP2IPRouting_ DestIPGroupName]	Defines the IP Group to where you want to route the call. The actual destination of the SIP dialog message depends on the IP Group type (as defined in the 'Type' parameter): ■ Server-type IP Group: The SIP dialog is sent to the IP address configured for the Proxy Set that is associated with the IP Group. ■ User-type IP Group: The device checks if the SIP dialog is from a registered user, by searching for a match between the Request-URI of the received SIP dialog and an AOR registration record in the device's database. If found, the device sends the SIP dialog to the IP address specified in the database for the registered contact. By default, no value is defined. **Note:**

Parameter	Description
	■ The parameter is applicable only if the 'Destination Type' parameter is configured to **IP Group**. However, you also need to specify this parameter if the 'Destination Type' parameter is configured to **Dest Address**, **Request URI**, **ENUM**, **Dial Plan** or **LDAP** (even though these calls are **not** sent to the specified IP Group). For these cases, it allows you to associate other configuration entities (such as an IP Profile) that are assigned to the IP Group, with the destination of these calls. ■ The selectable IP Group for the parameter depends on the assigned Routing Policy (in the 'Routing Policy' parameter in this table). For more information, see Configuring SBC Routing Policy Rules.
'Destination SIP Interface' `dest-sip-interface-name` [IP2IPRouting_ DestSIPInterfaceName]	Defines the destination SIP Interface to where the call is sent. By default, no value is defined. To configure SIP Interfaces, see Configuring SIP Interfaces. **Note:** ■ The parameter is applicable **only** if the 'Destination Type' parameter is configured to any value other than **IP Group**. If the 'Destination Type' parameter is configured to **IP Group**, the following SIP Interface is used: ✓ Server-type IP Groups: SIP Interface that is assigned to the Proxy Set associated with the IP Group. ✓ User-type IP Groups: SIP Interface is determined during user registration with the device. ■ For multi-tenancy, if the assigned Routing Policy is not shared (i.e., the Routing Policy is associated with an Isolated SRD), the SIP Interface must be one that is associated with the Routing Policy or with a shared Routing Policy (i.e., the Routing Policy is associated with one or more Shared SRDs). If the Routing Policy is shared, the SIP Interface can be one that is associated with any SRD or Routing Policy (but it's recommended that it belong to the same SRD/Routing Policy or to shared SRD/Routing Policy to avoid "bleeding").
'Destination Address' `dst-address` [IP2IPRouting_ DestAddress]	Defines the destination address to where the call is sent. The valid value is an IP address in dotted-decimal notation or an FQDN (domain name, e.g., domain.com). If ENUM-based routing is used (i.e., the 'Destination Type' parameter is set to **ENUM**) the parameter configures the address

Parameter	Description
	of the ENUM service, for example, e164.arpa, e164.customer.net or NRENum.net. The device sends the ENUM query containing the destination phone number to an external DNS server, configured in the IP Interfaces table. The ENUM reply includes a SIP URI (user@host) which is used as the destination Request-URI in this routing table. The valid value is a string of up to 50 characters (IP address or FQDN). By default, no value is defined. **Note:** ■ The parameter is applicable only if the 'Destination Type' parameter is configured to **Dest Address** [1] or **ENUM** [3]; otherwise, the parameter is ignored. ■ When using domain names, enter the DNS server's IP address or alternatively, define these names in the Internal DNS table (see Configuring the Internal SRV Table). ■ To terminate SIP OPTIONS messages at the device (i.e., to handle them locally), set the parameter to "internal".
'Destination Port' `dst-port` [IP2IPRouting_ DestPort]	Defines the destination port to where the call is sent.
'Destination Transport Type' `dst-transport-type` [IP2IPRouting_ DestTransportType]	Defines the transport layer type for sending the call. ■ [-1] = (Default) Not configured. The transport type is determined by the [SIPTransportType] global parameter. ■ [0] **UDP** ■ [1] **TCP** ■ [2] **TLS** ■ [3] **SCTP**
'IP Group Set' `ipgroupset-name` [IP2IPRouting_ IPGroupSetName]	Assigns an IP Group Set to the routing rule. The device routes the call to one of the IP Groups in the IP Group Set according to the load-balancing policy configured for the IP Group Set. For more information, see Configuring IP Group Sets. **Note:** The parameter is applicable only if you configure the 'Destination Type' parameter to **IP Group Set** (above).
'Call Setup Rules Set	Assigns a Call Setup Rule Set ID to the routing rule. The device

Parameter	Description
ID' `call-setup-rules-set-id` [IP2IPRouting_CallSetupRulesSetId]	performs the Call Setup rules of this Set ID if the incoming call matches the characteristics of this routing rule. The device routes the call to the destination according to the routing rule's configured action, only after it has performed the Call Setup rules. To configure Call Setup rules, see Configuring Call Setup Rules.
'Group Policy' `group-policy` [IP2IPRouting_GroupPolicy]	Defines whether the routing rule includes call forking. ■ [0] **None** = (Default) Call uses only this route (even if Forking Group members are configured in the rows below it). ■ [1] **Forking** = Call uses this route and the routes of Forking Group members, if configured (in the rows below it). **Note:** Each Forking Group can contain up to 20 members. In other words, up to 20 routing rules can be configured for the same Forking Group.
'Cost Group' `cost-group` [IP2IPRouting_CostGroup]	Assigns a Cost Group to the routing rule for determining the cost of the call. By default, no value is defined. To configure Cost Groups, see Configuring Cost Groups. **Note:** ■ To implement LCR and its Cost Groups, you must enable LCR for the Routing Policy assigned to the routing rule (see Configuring SBC Routing Policy Rules). If LCR is disabled, the device ignores the parameter. ■ The Routing Policy also determines whether matched routing rules that are **not** assigned Cost Groups are considered as a higher or lower cost route compared to matching routing rules that are assigned Cost Groups. For example, if the 'Default Call Cost' parameter in the Routing Policy is configured to **Lowest Cost**, even if the device locates matching routing rules that are assigned Cost Groups, the first-matched routing rule without an assigned Cost Group is considered as the lowest cost route and thus, chosen as the preferred route.
'Routing Tag Name' `routing-tag-name` [IP2IPRouting_RoutingTagName]	Defines the destination Dial Plan tag, which is used to determine the destination IP Group. The valid value is a string of up to 70 characters. Only **one** tag can be configured. Only the tag **name** must be configured (not the value, if exists). For example, if the tag is configured in the Dial Plan rule as "Country=England", configure the parameter to

Parameter	Description
	"Country" only. The tag is case insensitive. The default value is "default", meaning that the device uses the first tag name in the Dial Plan rule that is configured without a value. For example, if the Dial Plan rule is configured with tags "Country=England;City=London;Essex", the default tag is "Essex". For more information on using tags to determine destination IP Group, see Using Dial Plan Tags for Routing Destinations. **Note:** The parameter is applicable only if the 'Destination Type' parameter is configured to **Destination Tag** (see above).

'Internal Action'
`internal-action`
[IP2IPRouting_InternalAction]
	Defines a SIP response code (e.g., 200 OK) or a redirection response (with an optional Contact field indicating to where the sender must re-send the message) that the device sends to the sender of the incoming SIP dialog (instead of sending the call to another destination). The parameter is applicable only when the 'Destination Type' parameter in this table is configured to **Internal** (see above).
	The valid value syntax (case-insensitive) is:
	■ For SIP response codes:
	`reply(response='<code>')`
	The following example sends a SIP 200:
	`reply(response='200')`
	■ For redirection responses:
	`redirect(response='<code>',contact='sip:'+….)`
	`redirect(contact='…',response='<code>')`
	`redirect(contact='sip:user@host')`
	Examples:
	✓ The device responds to the dialog with a SIP 300 redirect response that includes a contact value:
	`redirect(response='300',contact='sip:102@host')`

Parameter	Description
	✓ The device redirects the call from the sender to a SIP Recording Server (SRS): `redirect (response='302',contact='sip:'+header.to.url.user+'@siprecording.com')` You can use the built-in syntax editor to help you configure the field. Click the **Editor** button located alongside the field to open the Editor, and then simply follow the on-screen instructions. **Note:** ■ The parameter can be used for normal and alternative routing. ■ The response code for redirect messages can only be 3xx.
'Modified Destination User Name' `modified-dest-user-name` [IP2IPRouting_ModifiedDestUserName]	Defines the user part of the Request-URI in the outgoing SIP dialog message. The valid value is a string of up to 60 characters. By default, no value is defined. **Note:** The parameter is currently used only when the device communicates with AudioCodes VoiceAI Connect voice-bot solution.

Configuring Rerouting of Calls to Fax Destinations

You can configure the device to reroute incoming SBC calls identified as fax calls to a new IP destination. The device identifies a fax call if it detects, within a user-defined interval, a calling (CNG) tone on the originator side (incoming IP leg). If the device detects a fax call, it terminates the call and reroutes it using a new INVITE to the new fax destination (new IP Group). If the initial INVITE that was used to establish the voice call was already sent, the device sends a CANCEL (if not connected yet) or a BYE (if already connected) to release the call (with the internal disconnect reason RELEASE_BECAUSE_FAX_REROUTING, translated to Q.850 reason GWAPP_NORMAL_UNSPECIFIED 31).

- You must configure the originating fax to use the G.711 coder.
- If the remote side replies with T.38 or G.711 VBD, fax rerouting is not done.
- If both fax rerouting and fax re-INVITE are configured, only fax rerouting is done.

The following provides a basic example on how to configure fax rerouting.

➢ **To configure fax rerouting:**

1. Open the Fax/Modem/CID Settings page (**Setup** menu > **Signaling & Media** tab > **Media** folder > **Fax/Modem/CID Settings**).
   a. In the 'Fax Detection Timeout' field [SBCFaxDetectionTimeout], enter the duration (in seconds) for which the device attempts to detect fax (CNG tone):

   b. From the 'CNG Detector Mode' drop-down list [CNGDetectorMode], select **Event Only**.
2. Load an ini file to the device through the Auxiliary Files page (see Loading Auxiliary Files through Web Interface on page 863) with the following parameter setting, which enables in-band network detection related to fax:

```
EnableFaxModemInbandNetworkDetection = 1
```

3. In the IP Groups table (see Configuring IP Groups), configure the following IP Groups:
   - IP Group #0 "HQ": This is the source IP Group, sending voice calls and fax calls.
   - IP Group #1 "Voice": This is the destination for voice calls sent from IP Group #0.
   - IP Group #2 "Fax": This is the destination for fax calls sent from IP Group #0.
4. For the fax destination (IP Group #2), do the following:
   a. In the Coder Groups table (see Configuring Coder Groups), configure a Coder Group with T.38 to enable fax transmission over IP.
   b. In the IP Profiles table (see Configuring IP Profiles), configure an IP Profile:
      i. From the 'Fax Coders Group' drop-down list, select the Coder Group that you configured above.
      ii. From the 'Fax Mode' drop-down list, select **Handle always**.
   c. In the IP Groups table, edit IP Group #2, and then from the 'IP Profile' drop-down list, select the IP Profile that you configured above.
5. For the voice destination (IP Group #1), do the following:
   a. In the IP Profiles table, configure an IP Profile - from the 'Fax Rerouting Mode' drop-down list, select **Rerouting without delay**:

   b. In the IP Groups table, edit IP Group #1, and then from the 'IP Profile' drop-down list, select the IP Profile that you configured above.
6. In the IP-to-IP Routing table (see Configuring SBC IP-to-IP Routing Rules), configure the following adjacent rows of IP-to-IP Routing rules:

- IP-to-IP Routing Rule #0 to route voice calls from IP Group #0 to IP Group #1:

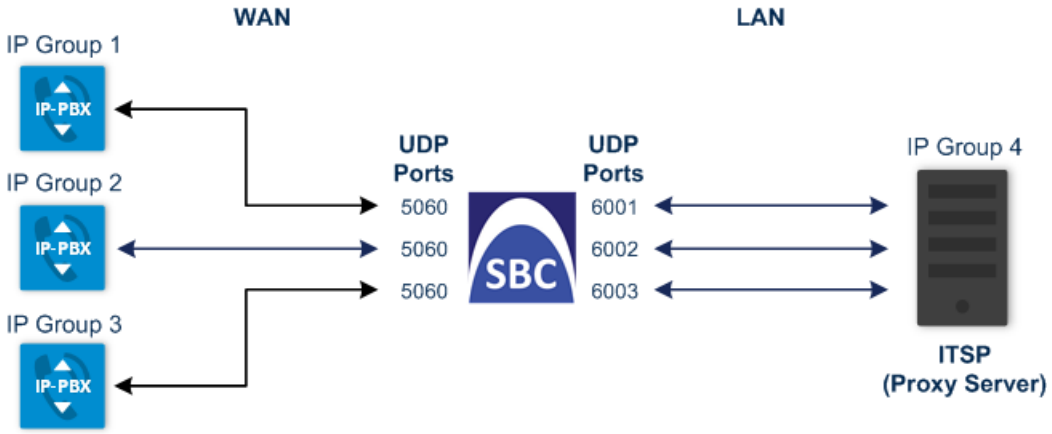
Match	
Source IP Group	**HQ** (IP Group #0)
Call Trigger	**Initial Only**
ReRoute IP Group	**Voice** (IP Group #1)
Action	
Destination Type	**IP Group**
Destination IP Group	**Voice** (IP Group #1)

- IP-to-IP Routing Rule #1 to route fax calls from IP Group #0 to IP Group #2:

Match	
Source IP Group	**HQ** (IP Group #0)
Call Trigger	**Fax Rerouting**
Action	
Destination Type	**IP Group**
Destination IP Group	**Fax** (IP Group #2)

Configuring Specific UDP Ports using Tag-based Routing

You can configure the device to use a specific local UDP port for each SIP entity (e.g., PBX) communicating with a common proxy server (e.g., ITSP). The figure below illustrates an example scenario of such an implementation, whereby the device uses a specific local UDP port (e.g., 6001, 6002, and 6003) for each IP PBX, on the leg interfacing with the proxy server:



For each IP PBX, the device sends SIP messages to the proxy server using the specific local, UDP port on the leg interfacing with the proxy server. For SIP messages received from the proxy server, the device routes the messages to the appropriate IP PBX according to the local UDP port on which the message was received. On the leg interfacing with the IP PBXs, the device uses the same local UDP port (e.g., 5060) for all IP PBXs (send and receive).

To configure this feature, you need to configure the SIP Interface of the proxy server with a special UDP port range, and use tag-based routing with Call Setup Rules to specify the exact UDP port you want assigned to each SIP entity (IP PBX), from the SIP Interface port range. The following procedure describes how to configure the device to use a specific local UDP port per SIP entity on the leg interfacing with a proxy server that is common to all the SIP entities. To facilitate understanding, the procedure is based on the previous example.

➢ **To configure specific UDP ports for SIP entities communicating with common proxy server:**

1. Open the SIP Interfaces table (see Configuring SIP Interfaces), and then configure the following SIP Interfaces:
   - SIP Interface for leg interfacing with IP PBXs (local UDP port 5060 is used):

General	
Index	**1**
Name	**PBX**
Network Interface	**WAN**
UDP Port	**5060**

   - SIP Interface for leg interfacing with proxy server (specific local UDP ports are later taken from this port range):

General	
Index	**2**
Name	**ITSP**
Network Interface	**LAN**
UDP Port	**5060**
Additional UDP Ports	**6000-7000**

For guidelines on configuring the 'Additional UDP Ports' parameter (SIPInterface_AdditionalUDPPorts), see Configuring SIP Interfaces.

2. Open the IP Groups table (see Configuring IP Groups), and then configure the following IP Groups:

- IP Group for the first IP PBX ("Type" and "Port" tags are later used to identify the IP PBX and assign it a local UDP port 6001 on the leg interfacing with the proxy server):

General	
Index	**1**
Name	**PBX-1**
Type	**Server**
SBC Advanced	
Call Setup Rules Set ID	**1**
Tags	**Type=PBX;Port=6001**

- IP Group for the second IP PBX ("Type" and "Port" tags are later used to identify the IP PBX and assign it a local UDP port 6002 on the leg interfacing with the proxy server):

General	
Index	**2**
Name	**PBX-2**
Type	**Server**
SBC Advanced	
Call Setup Rules Set ID	**1**
Tags	**Type=PBX;Port=6002**

- IP Group for the third IP PBX ("Type" and "Port" tags are later used to identify the IP PBX and assign it a local UDP port 6003 on the leg interfacing with the proxy server):

General	
Index	**3**

General	
Name	**PBX-3**
Type	**Server**
SBC Advanced	
Call Setup Rules Set ID	**1**
Tags	**Type=PBX;Port=6003**

- IP Group for the proxy server ("Type" tag is later used to identify proxy server):

General	
Index	**4**
Name	**ITSP**
Type	**Server**
SBC Advanced	
Call Setup Rules Set ID	**1**
Tags	**Type=ITSP**

3. Open the Call Setup Rules table (see Configuring Call Setup Rules), and then configure the following Call Setup rules:
   - Uses the value of the "Type" tag name, configured in the IP Group's 'Tags' parameter, as the source tag:

General	
Index	**1**
Rule Set ID	**1**
Action	
Action Subject	**srctags.Type**
Action Type	**Modify**
Action Value	**param.ipg.src.tags.Type**

- If the source tag name "Type" equals "PBX" (i.e., SIP message from an IP Group belonging to one of the IP PBXs), then use the value of the "Port" tag name, configured in the 'Tags' parameter of the classified IP Group, as the local UDP port on the leg interfacing with the proxy server for messages sent to the proxy server:

General	
Index	**2**
Rule Set ID	**1**
Condition	**srctags.Type=='PBX'**
Action	
Action Subject	**message.outgoing.local-port**
Action Type	**Modify**
Action Value	**param.ipg.src.tags.Port**

- If the source tag name "Type" equals "ITSP" (i.e., SIP message from the ITSP), then use the value (port number) of the local port on which the incoming message from the proxy server is received by the device, as the value of the destination tag name "Port". In other words, the value could either be "6001", "6002", or "6003". This value is then used by the IP-to-IP Routing table to determine to which IP PBX to send the message. For example, if the destination tag value is "6001", the device identifies the destination as "PBX-1":

General	
Index	**3**
Rule Set ID	**1**
Condition	**srctags.Type=='ITSP'**
Action	
Action Subject	**dsttags.Port**
Action Type	**Modify**
Action Value	**message.incoming.local-port**

4. Open the IP-to-IP Routing table (see Configuring SBC IP-to-IP Routing Rules), and then configure the following IP-to-IP Routing rules:

- Routes calls from the IP PBXs (identified by the source tag name-value "Type=PBX") to the ITSP (identified as an IP Group):

General	
Index	**1**
Name	**PBX-to-ITSP**
Match	
Source Tag	**Type=PBX**
Action	
Destination Type	**IP Group**
Destination IP Group	**ITSP**

- Routes calls from the ITSP (identified by the source tag name-value "Type=ITSP") to the IP PBXs (identified by the specific port assigned to the IP PBX by the value of the destination tag name "Port"):

General	
Index	**2**
Name	**ITSP-to-PBX**
Match	
Source Tag	**Type=ITSP**
Action	
Destination Type	**Destination Tag**
Routing Tag Name	**Port**

Configuring a Routing Response Timeout

If you have routing rules in the IP-to-IP Routing table that need to query external servers (e.g., LDAP server, ENUM server, or HTTP GET method requests) on whose responses the device uses to determine where to route the SBC calls, you can configure a timeout for the responses. If the timeout expires before the device receives a response, the device sends a routing failure message (SIP 500) to the caller or uses an alternative routing rule (if configured).

➢ **To configure a timeout for routing query responses:**

1. Open the SBC General Settings page (**Setup** menu > **Signaling & Media** tab > **SBC** folder > **SBC General Settings**).
2. In the 'Routing Timeout' [SbcRoutingTimeout] field, enter the maximum duration (in seconds) that the device is prepared to wait for a response from external servers.

3. Click **Apply**.

Configuring SIP Response Codes for Alternative Routing Reasons

The Alternative Reasons Set table lets you configure groups of SIP response codes for SBC call release (termination) reasons that trigger alternative routing. This feature works together with the Proxy Hot Swap feature, which is configured in the Proxy Sets table.

Alternative routing based on SIP responses is configured using two tables with "parent-child" relationship:

- **Alternative Reasons Set table ("parent"):** Defines the name of the Alternative Reasons Set. You can configure up to 10 Alternative Reasons Sets.
- **Alternative Reasons Rules table ("child"):** Defines SIP response codes per Alternative Reasons Set. You can configure up to 200 Alternative Reasons Rules in total, where each Alternative Reasons Set can include up to 20 Alternative Reasons Rules.

To apply your configured alternative routing reason rules, you need to assign the Alternative Reasons Set for which you configured the rules, to the relevant IP Group in the IP Groups table, using the 'SBC Alternative Routing Reasons Set' parameter.

In addition to configuring the response codes described in this section, you need to configure the following:

- A Proxy Set with one or more addresses (proxy servers) and whose 'Proxy Hot Swap' parameter is configured to **Enable** (see Configuring Proxy Sets).
- An IP-to-IP Routing rule 1) whose 'Destination IP Group' parameter is a Server-type IP Group that is associated with the above Proxy Set (see Configuring SBC IP-to-IP Routing Rules) and 2) that is assigned the relevant Alternative Reasons Set (using the 'SBC Alternative Routing Reasons Set' parameter).
- An alternative IP-to-IP Routing rule for the above rule.

Alternative routing based on SIP response codes operates as follows:

1. The device sends (outgoing) a SIP dialog-initiating message (e.g., INVITE, OPTIONS, and SUBSCRIBE) to one of the online proxy servers (addresses) configured for the Proxy Set that is associated with the destination IP Group of the matched IP-to-IP Routing rule.

2. If there is no response to the sent SIP message, or a "reject" (release) response is received (e.g., SIP 406) that is also configured for the Alternative Reasons Set assigned to the destination IP Group, the device tries to route the SIP message again (re-transmission) to the same proxy for a user-defined number of times, configured by the [HotSwapRtx] parameter. If still unsuccessful, the device tries to send the message to a different online proxy of the Proxy Set and if unsuccessful, it tries another online proxy, and so on. The order of attempted online proxies is according to the Proxy Set's configuration.

   The following can then occur depending on received response codes or no responses:

   - If any attempted proxy sends a response code that you have not configured for the assigned Alternative Reasons Set, the routing of the SIP message fails and the device **does not** make any further attempts to route the message.
   - If the device has tried all the online proxies of the Proxy Set and no response has been received or responses have been received that you have also configured for the assigned Alternative Reasons Set, the device searches the IP-to-IP Routing table for a matching alternative routing rule and if found, sends the SIP message to the destination configured for that alternative routing rule (repeating steps 1 through 2 above, if needed).

You can also configure alternative routing for the following proprietary response codes (if configured in the table) that are issued by the device itself:

- **806 Media Limits Exceeded:** The device generates the response code when the call is terminated due to crossed user-defined thresholds of QoE metrics such as MOS, packet delay, and packet loss (see Configuring Quality of Experience Profiles) and/or media bandwidth (see Configuring Bandwidth Profiles). When this occurs, the device sends a SIP 480 (Temporarily Unavailable) response to the SIP entity (IP Group). This is configured by 1) assigning an IP Group a QoE and/or Bandwidth profile that rejects calls if the threshold is crossed, 2) configuring 806 for an Alternative Reasons Set that is assigned to the IP Group and 3) configuring an alternative routing rule.

  The device also generates the response code when it rejects a call based on Quality of Service rules due to crossed Voice Quality and Bandwidth thresholds (see Configuring Quality of Service Rules). If the response code is configured in the table and the device rejects a call due to threshold crossing, it searches in the IP-to-IP Routing table for an alternative routing rule.

- **850 Signalling Limits Exceeded:** The device generates the response code when it rejects a call based on Quality of Service rules due to crossed ASR, NER or ACD thresholds (see Configuring Quality of Service Rules). If the response code is configured for an Alternative Reasons Set that is assigned to the IP Group and the device rejects a call due to threshold crossing, it searches in the IP-to-IP Routing table for an alternative routing rule.

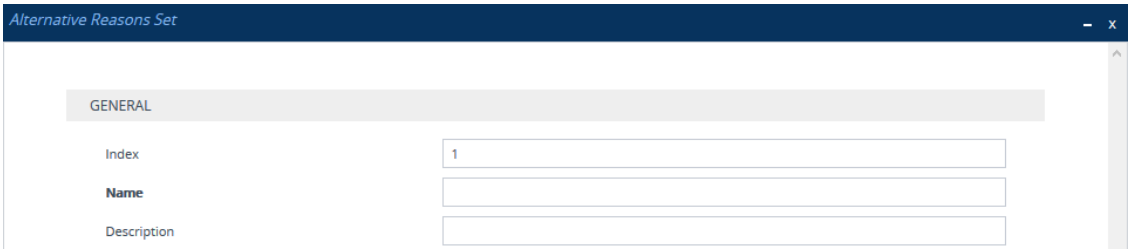
- The device issues itself the SIP response code 408 when no response is received from a sent SIP message.
- If the device receives a SIP 408 response, an ICMP message, or no response, it still does alternative routing even if you have not configured this code for an Alternative Reasons Set.
- SIP requests belonging to an SRD or IP Group that have reached the call limit (maximum concurrent calls and/or call rate), configured in the Call Admission table, are sent to an alternative route (if configured in the IP-to-IP Routing table for the SRD or IP Group). If no alternative routing rule is found, the device automatically rejects the SIP request with a SIP 480 (Temporarily Unavailable) response.
- If due to an INVITE message the device receives from the proxy a SIP 18x response (e.g., 180 or 183) followed by any failure response (e.g., 400 Not Found), the device does not do alternative routing, but instead terminates the call. This occurs even if the failure response is configured in the associated Alternative Reasons Set.

The following procedure describes how to configure Alternative Reasons sets through the Web interface. You can also configure it through other management platforms:

- Alternative Reasons Set table: ini file [SBCAltRoutingReasonsSet] or CLI (`configure voip > sbc routing alt-route-reasons-set`)
- Alternative Reasons Rules table: ini file [SBCAltRoutingReasonsList] or CLI (`configure voip > sbc routing alt-route-reasons-set < alt-route-reasons-rules`)

➢ To configure SIP reason codes for alternative IP routing:

1. Open the Alternative Reasons Set table (**Setup** menu > **Signaling & Media** tab > **SBC** folder > **Routing** > **Alternative Reasons Set**).
2. Click **New**; the following dialog box appears:

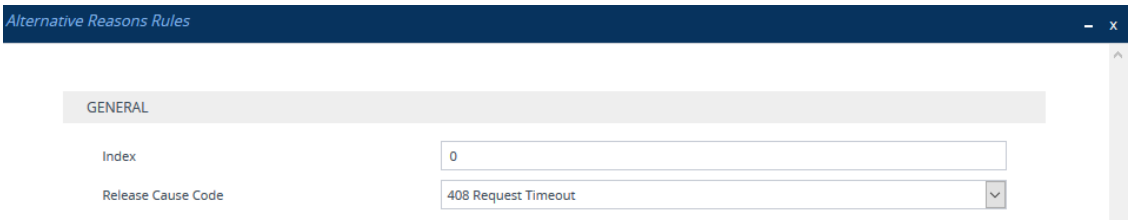


3. Configure an Alternative Reasons Set according to the parameters described in the table below.
4. Click **Apply**.

Table 25-3: Alternative Reasons Set Table Parameter Descriptions

Parameter	Description
'Index' [SBCAltRoutingReasonsSet_Index]	Defines an index number for the new table row. **Note:** Each row must be configured with a unique index.
'Name' `name` [SBCAltRoutingReasonsSet_Name]	Defines an arbitrary name to easily identify the row. The valid value is a string of up to 40 characters. **Note:** Each row must be configured with a unique name.
'Description' `description` [SBCAltRoutingReasonsSet_Description]	Defines a description for the Alternative Reasons Set. The valid value is a string of up to 99 characters. By default, no value is defined.

5. Select the index row of the Alternative Reasons Set that you added, and then click the **Alternative Reasons Rules** link located at the bottom of the page; the Alternative Reasons Rules table opens.
6. Click **New**; the following dialog box appears:



7. Configure Alternative Reasons rules according to the parameters described in the table below.
8. Click **Apply**.

Table 25-4: Alternative Reasons Rules Table Parameter Descriptions

Parameter	Description
'Index' `alt-route-reasons-rules` [SBCAltRoutingReasonsList_	Defines an index number for the new table row. **Note:** Each row must be configured with a unique index.

Parameter	Description
SBCAltRouteIndex]	
'Release Cause Code' `rel-cause-code` [SBCAltRoutingReasonsList_ReleaseCauseCode]	Defines a SIP response code that triggers the device's alternative routing mechanism. [4] **4xx**; [5] **5xx**; [6] **6xx**; [400] **400 Bad Request**; [402] **402 Payment Required**; [403] **403 Forbidden**; [404] **404 Not Found**; [405] **405 Method Not Allowed**; [406] **406 Not Acceptable**; [408] **408 Request Timeout** (Default); [409] **409 Conflict**; [410] **410 Gone**; [413] **413 Request Too Large**; [414] **414 Request URI Too Long**; [415] **415 Unsupported Media**; [420] **420 Bad Extension**; [421] **421 Extension Required**; [423] **423 Session Interval Too Small**; [480] **480 Unavailable**; [481] **481 Transaction Not Exist**; [482] **482 Loop Detected**; [483] **483 Too Many Hops**; [484] **484 Address Incomplete**; [485] **485 Ambiguous**; [486] **486 Busy**; [487] **487 Request Terminated**; [488] **488 Not Acceptable Here**; [491] **491 Request Pending**; [493] **493 Undecipherable**; [500] **500 Internal Error**; [501] **501 Not Implemented**; [502] **502 Bad Gateway**; [503] **503 Service Unavailable**; [504] **504 Server Timeout**; [505] **505 Version Not Supported**; [513] **513 Message Too Large**; [600] **600 Busy Everywhere**; [603] **603 Decline**; [604] **604 Does Not Exist Anywhere**; [606] **606 Not Acceptable**; [806] **806 Media Limits Exceeded**; [850] **850 Signalling Limits Exceeded**.

Configuring SBC Routing Policy Rules

The Routing Policies table lets you configure up to 250 Routing Policy rules. A Routing Policy determines the routing and manipulation (inbound and outbound) rules per SRD in a multiple SRD configuration topology. The Routing Policy also configures the following:

- Enables Least Cost Routing (LCR), and configures default call cost (highest or lowest) and average call duration for routing rules that are not assigned LCR Cost Groups. The default call cost determines whether matched routing rules that are not assigned Cost Groups are considered as a higher or lower cost route compared to other matching routing rules that are assigned Cost Groups. If you disable LCR, the device ignores the Cost Groups assigned to the routing rules in the IP-to-IP Routing table.
- Assigns LDAP servers (LDAP Server Group) for LDAP-based routing. IP-to-IP routing rules configured for LDAP or CSR (Call Setup Rules) queries use the LDAP server(s) that is assigned to the routing rule's associated Routing Policy. You can configure a Routing Policy per SRD or alternatively, configure a single Routing Policy that is shared between all SRDs.

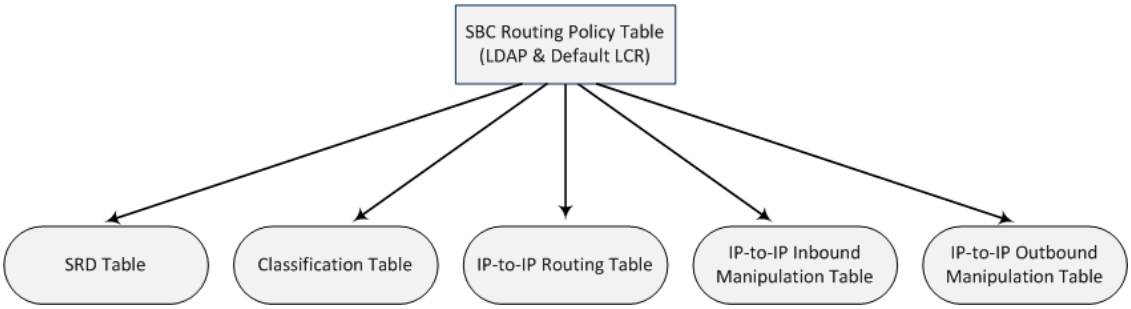
The implementation of Routing Policies is intended for the following deployments **only**:

- Deployments requiring LCR and/or LDAP-based routing.
- Multi-tenant deployments that require multiple, logical routing tables where each tenant has its own dedicated ("separated") routing (and manipulation) table. In such scenarios, each SRD (tenant) is configured as an Isolated SRD and assigned its own unique Routing Policy, implementing an almost isolated, non-bleeding routing configuration topology.

For all other deployment scenarios, the Routing Policy is irrelevant and the handling of the configuration entity is not required as a default Routing Policy ("Default_SBCRoutingPolicy" at Index 0) is provided. When only one Routing Policy is required, the device automatically associates the default Routing Policy with newly added configuration entities that can be associated with the Routing Policy (as mentioned later in this section, except for Classification rules). This facilitates configuration, eliminating the need to handle the Routing Policy configuration entity (except if you need to enable LCR and/or assign an LDAP server to the Routing Policy). In such a setup, where only one Routing Policy is used, single routing and manipulation tables are employed for all SRDs.

> If possible, it is recommended to use only **one** Routing Policy for all SRDs (tenants), unless deployment requires otherwise (i.e., a dedicated Routing Policy per SRD).

Once configured, you need to associate the Routing Policy with an SRD(s) in the SRDs table. To determine the routing and manipulation rules for the SRD, you need to assign the Routing Policy to routing and manipulation rules. The figure below shows the configuration entities to which Routing Policies can be assigned:



Typically, assigning a Routing Policy to a Classification rule is not required, as when an incoming call is classified it uses the Routing Policy associated with the SRD to which it belongs. However, if a Routing Policy is assigned to a Classification rule, it overrides the Routing Policy assigned to the SRD. The option to assign Routing Policies to Classification rules is useful in deployments requiring different routing and manipulation rules for specific calls pertaining to the **same** SRD. In such scenarios, you need to configure multiple Classification rules for the same SRD, where for some rules no Routing Policy is assigned (i.e., the SRD's assigned Routing Policy is used) while for others a different Routing Policy is specified to override the SRD's assigned Routing Policy.

In multi-tenant environments employing multiple SRDs and Routing Policies, the IP Groups that can be used in routing rules (in the IP-to-IP Routing table) are as follows:

- If the Routing Policy is assigned to only one SRD and the SRD is an Isolated SRD, the routing rules of the Routing Policy can be configured with IP Groups belonging to the Isolated SRD and IP Groups belonging to all Shared SRDs.
- If the Routing Policy is assigned to a Shared SRD, the routing rules of the Routing Policy can be configured with any IP Group (i.e., belonging to Shared and Isolated SRDs). In effect, the Routing Policy can include routing rules for call routing between Isolated SRDs.
- If the Routing Policy is assigned to multiple SRDs (Shared and/or Isolated), the routing rules of the Routing Policy can be configured with IP Groups belonging to all Shared SRDs as well as IP Groups belonging to Isolated SRDs that are assigned the Routing Policy.

To facilitate the configuration of routing rules in the IP-to-IP Routing table through the Web interface, only the permitted IP Groups (according to the above) are displayed as optional values.

The general flow for processing the call for multi-tenant deployments and Routing Policies is as follows:

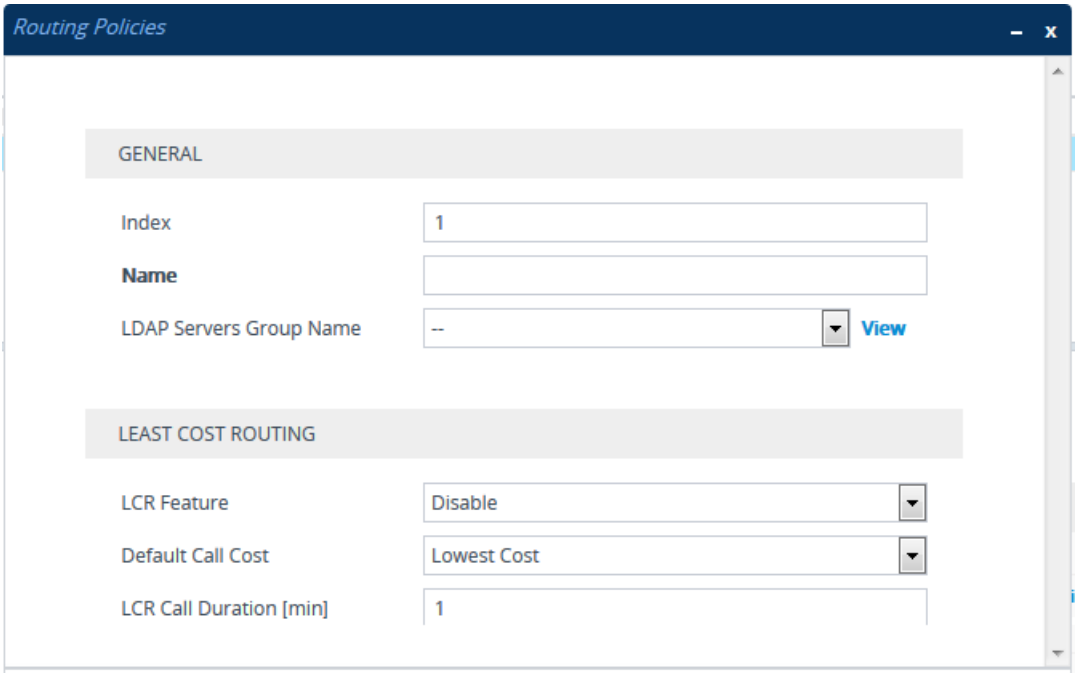
1. Using the Classification table, the device classifies the incoming call to an IP Group, based on the SIP Interface on which the call is received. Based on the SIP Interface, the device associates the call to the SRD that is assigned to the SIP Interface.
2. Once the call has been successfully classified to an IP Group, the Routing Policy assigned to the associated SRD is used. However, if a Routing Policy is configured in the Classification table, it overrides the Routing Policy assigned to the SRD.
3. The regular manipulation (inbound and outbound) and routing processes are done according to the associated Routing Policy.

- The Classification table is used only if classification by registered user in the device's users registration database or by Proxy Set fails.
- If the device receives incoming calls (e.g., INVITE) from users that have already been classified and registered in the device's registration database, the device ignores the Classification table and uses the Routing Policy that was determined for the user during the initial classification process.

The following procedure describes how to configure Routing Policies rules through the Web interface. You can also configure it through ini file [SBCRoutingPolicy] or CLI (`configure voip > sbc routing sbc-routing-policy`).

➢ **To configure a Routing Policy rule:**

1. Open the Routing Policies table (**Setup** menu > **Signaling & Media** tab > **SBC** folder > **Routing** > **Routing Policies**).
2. Click **New**; the following dialog box appears:



3. Configure the Routing Policy rule according to the parameters described in the table below.
4. Click **Apply**.

Table 25-5: Routing Policies table Parameter Descriptions

Parameter	Description
General	
'Index'	Defines an index number for the new table row. **Note:** Each row must be configured with a unique index.
'Name' `name` [SBCRoutingPolicy_Name]	Defines a descriptive name, which is used when associating the row in other tables. The valid value is a string of up to 40 characters. By default, no name is defined. If you don't configure a name, the device automatically assigns a name in the following format: "SBCRoutingPolicy_<Index>", for example, "SBCRoutingPolicy_2". **Note:** ■ Each row must be configured with a unique name. ■ The parameter value cannot contain a forward slash (/).
'LDAP Servers Group Name' `ldap-srv-group-name` [SBCRoutingPolicy_	Assigns an LDAP Server Group to the Routing Policy. Routing rules in the IP-to-IP Routing table that are associated with the Routing Policy and that are configured with LDAP and/or Call Setup Rules, use the LDAP server(s) configured for this LDAP Server Group.

Parameter	Description
LdapServersGroupName]	By default, no value is defined. For more information on LDAP Server Groups, see Configuring LDAP Server Groups. **Note:** The default Routing Policy is assigned the default LDAP Server Group ("DefaultCTRLServersGroup").
Least Cost Routing	
'LCR Feature' `lcr-enable` [SBCRoutingPolicy_ LCREnable]	Enables the Least Cost Routing (LCR) feature for the Routing Policy. ■ [0] **Disable** (default) ■ [1] **Enable** For more information on LCR, see Least Cost Routing.
'Default Call Cost' `lcr-default-cost` [SBCRoutingPolicy_ LCRDefaultCost]	Defines whether routing rules in the IP-to-IP Routing table that are not assigned a Cost Group are considered a higher cost or lower cost route compared to other matched routing rules that are assigned Cost Groups. ■ [0] **Lowest Cost** = (Default) The device considers a matched routing rule (belonging to the Routing Policy) that is not assigned a Cost Group as the lowest cost route. Therefore, it uses the routing rule. ■ [1] **Highest Cost** = The device considers a matched routing rule (belonging to the Routing Policy) that is not assigned a Cost Group as the highest cost route. Therefore, it is only used if the other matched routing rules that are assigned Cost Groups are unavailable. **Note:** If multiple matched routing rules without an assigned Cost Group exist, the device selects the first matched rule in the table.
'LCR Call Duration' `lcr-call-length` [SBCRoutingPolicy_ LCRAverageCallLength]	Defines the average call duration (in minutes) and is used to calculate the variable portion of the call cost. This is useful, for example, when the average call duration spans over multiple time bands. The LCR is calculated as follows: cost = call connect cost + (minute cost * average call duration). The valid value is 0-65533. The default is 1. For example, assume the following Cost Groups: ■ "Weekend A": call connection cost is 1 and charge per

Parameter	Description
	minute is 6. Therefore, a call of 1 minute cost 7 units. ■ "Weekend B": call connection cost is 6 and charge per minute is 1. Therefore, a call of 1 minute cost 7 units. Therefore, for calls under one minute, "Weekend A" carries the lower cost. However, if the average call duration is more than one minute, "Weekend B" carries the lower cost.

Configuring IP Group Sets

The IP Group Set table lets you configure up to 312 IP Group Sets. An IP Group Set is a group of IP Groups used for load balancing of calls, belonging to the same source, to a call destination (i.e., IP Group). Each IP Group Set can include up to five IP Groups (Server-type and Gateway-type only). The chosen destination IP Group for each call depends on the configured load-balancing policy, which can be Round Robin, Random Weights, or Homing (for more information, see the table's description, later in this section).

Alternative routing within the IP Group Set is also supported. If a chosen destination IP Group responds with a reject response that is configured for an Alternative Reasons Set (see Configuring SIP Response Codes for Alternative Routing Reasons) that is assigned to the IP Group ('SBC Alternative Routing Reasons Set' parameter), or doesn't respond at all (i.e., keep-alive with its' associated Proxy Set fails), the device attempts to send the call to another IP Group in the IP Group Set (according to the load-balancing policy). For enabling Proxy Set keep-alive, see Configuring Proxy Sets.

An example of round-robin load-balancing and alternative routing: The first call is sent to IP Group #1 in the IP Group Set, the second call to IP Group #2, and the third call to IP Group #3. If the call sent to IP Group #1 is rejected, the device employs alternative routing and sends it to IP Group #4.

Once you have configured your IP Group Set, to implement call load-balancing by IP Groups, do one of the following:

- In the IP-to-IP Routing table, configure the routing rule's 'Destination Type' parameter to **IP Group Set**, and then assign it the IP Group Set in the 'IP Group Set' parameter.
- If you are routing to IP Groups based on Dial Plan tags:
  - In the IP Group Set table (see below), specify the tag name.
  - In the IP-to-IP Routing table, configure the routing rule's 'Destination Type' parameter to **Destination Tag**, and then specify the tag name in the 'Routing Tag Name' parameter.

For more information on IP-to-IP Routing rules, see Configuring SBC IP-to-IP Routing Rules. For more information on routing based on destination Dial Plan tags, see Using Dial Plan Tags for Routing Destinations.

IP Group Sets are configured using two tables with parent-child type relationship:

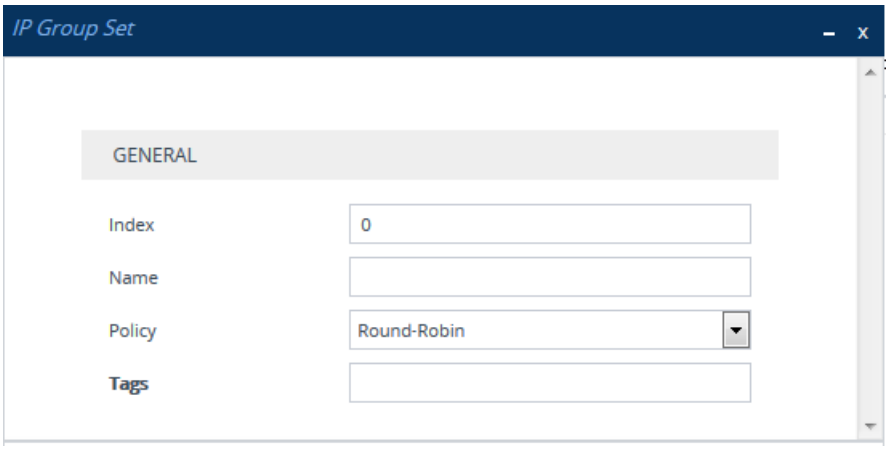
- **Parent table:** IP Group Set table, which defines the name and load-balancing policy of the IP Group Set.
- **Child table:** IP Group Set Member table, which assigns IP Groups to IP Group Sets. You can assign up to five IP Groups per IP Group Set.

The following procedure describes how to configure IP Group Sets through the Web interface. You can also configure it through other management platforms:

- **IP Group Set Table:** *ini* file [IPGroupSet] or CLI (`configure voip > sbc routing ip-group-set`)
- **IP Group Set Member Table:** *ini* file [IPGroupSetMember] or CLI (`configure voip > sbc routing ip-group-set-member`)

➢ To configure an IP Group Set:

1. Open the IP Group Set table (**Setup** menu > **Signaling & Media** tab > **SBC** folder > **Routing** > **IP Group Set**).
2. Click **New**; the following dialog box appears:



3. Configure the IP Group Set according to the parameters described in the table below.
4. Click **Apply**.

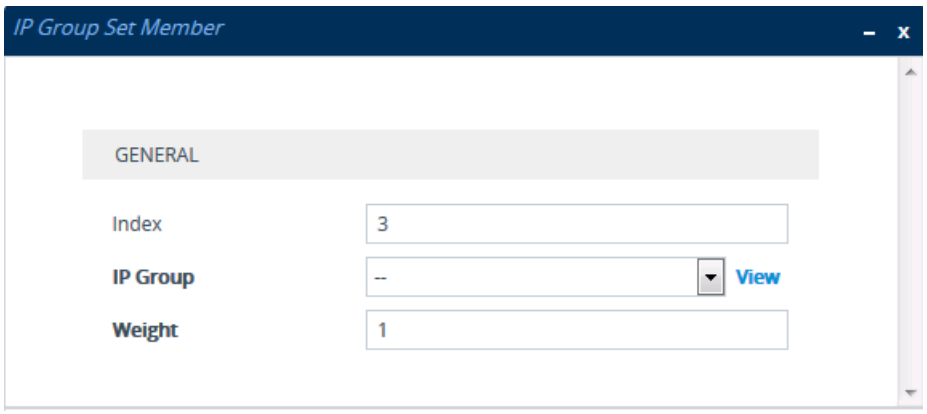
Table 25-6: IP Group Set Table Parameter Descriptions

Parameter	Description
'Index' [IPGroupSet_Index]	Defines an index number for the new table row. **Note:** Each row must be configured with a unique index.
'Name' `name` [IPGroupSet_	Defines a descriptive name, which is used when associating the row in other tables. **Note:**

Parameter	Description
Name]	■ Each row must be configured with a unique name. ■ The parameter value cannot contain a forward slash (/).
'Policy' policy [IPGroupSet_Policy]	Defines the load-balancing policy. ■ [0] **Round-Robin** = (Default) The device selects the next consecutive, available IP Group for each call. The device selects the first IP Group in the table (i.e., lowest index) for the first call and the next consecutive IP Groups for the next calls. For example, first call to IP Group at Index 0, second call to IP Group at Index 2, third call to IP Group at Index 3, and so on. If an IP Group is offline, the device selects the next consecutive IP Group. Once the last IP Group in the IP Group Set list is selected for a call, the device goes to the beginning of the list and sends the next call to the first IP Group, and so on. ■ [1] **Random Weight** = The device selects IP Groups at random and their weights determine their probability of getting chosen over others. The higher the weight, the more chance of the IP Group being chosen. ■ [2] **Homing** = The device always attempts to send all calls to the first IP Group in the table (i.e., lowest index). If unavailable, it sends the calls to the next consecutive, available IP Group. However, if the first IP Group comes online again, the device selects it. **Note:** For the **Random Weight** optional value, use the 'Weight' parameter in the IP Group Set Member table (below) to configure weight value per IP Group.
'Tags' tags [IPGroupSet_Tags]	Assigns a Dial Plan tag that is used to determine whether the incoming SIP dialog is sent to IP Groups belonging to this IP Group Set. The parameter is used when IP-to-IP Routing rules are configured for destination based on tags (i.e., 'Destination Type' parameter configured to Destination Tag). For more information on routing based on destination tags, see Using Dial Plan Tags for Routing Destinations. The valid value is a string of up to 70 characters. By default, no value is defined. You can configure the parameter with up to five tags, where each tag is separated by a semicolon (;). However, you can configure only up to four tags containing a name and value (e.g., Country=Ireland), and **one** tag containing a value only (e.g., Ireland). You can also configure multiple tags with the same name (e.g., Country=Ireland;Country=Scotland). The following example configures the maximum number of tags (i.e., four name=value tags and one value-only tag):

Parameter	Description
	Country=Ireland;Country=Scotland;Country=RSA;Country=Canada;USA. **Note:** If the IP Groups belonging to the IP Group Set are also configured with Dial Plan tags, the Dial Plan tag configured for the parameter takes precedence. If the same Dial Plan tag is also configured for other IP Groups in the IP Groups table, the IP Group Set takes precedence and the device sends the SIP dialog to the IP Group(s) belonging to the IP Group Set.

5. Select the IP Group Set row for which you want to assign IP Groups, and then click the **IP Group Set Member** link located below the table; the IP Group Set Member table appears.
6. Click **New**; the following dialog box appears:



7. Configure IP Group Set members according to the parameters described in the table below.
8. Click **Apply**, and then save your settings to flash memory.

Table 25-7: IP Group Set Member Table Parameter Descriptions

Parameter	Description
'Index' `index` [IPGroupSetMember_ IPGroupSetMemberIndex]	Defines an index number for the new table row. **Note:** Each row must be configured with a unique index.
'IP Group' `ip-group-name` [IPGroupSetMember_ IPGroupName]	Assigns an IP Group to the IP Group Set. To configure IP Groups, see Configuring IP Groups. **Note:** The IP Group can only be a Server-type or Gateway-type.
'Weight' `weight` [IPGroupSetMember_	Defines the weight of the IP Group. The higher the weight, the more chance of the IP Group being selected as the destination of the call.

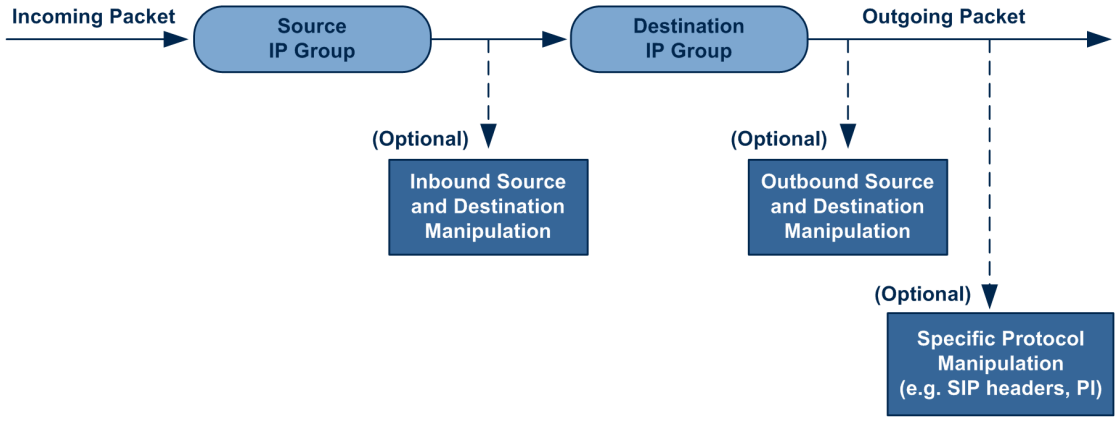
Parameter	Description
Weight]	The valid value is 1 to 9. The default is 1. **Note:** The parameter is applicable only if you configure the 'Policy' parameter to **Random Weight**.

26 SBC Manipulations

This section describes the configuration of the manipulation rules for the SBC application.

For additional manipulation features, see the following:
- Configuring SIP Message Policy Rules
- Configuring SIP Message Manipulation

The device supports SIP URI user part (source and destination) manipulations for inbound and outbound routing. These manipulations can be applied to a source IP group, source and destination host and user prefixes, and/or user-defined SIP request (e.g., INVITE, OPTIONS, SUBSCRIBE, and/or REGISTER). Since outbound manipulations are performed after routing, the outbound manipulation rule matching can also be done by destination IP Group. Manipulated destination user and host are performed on the following SIP headers: Request-URI, To, and Remote-Party-ID (if exists). Manipulated source user and host are performed on the following SIP headers: From, P-Asserted (if exists), P-Preferred (if exists), and Remote-Party-ID (if exists).



You can also restrict source user identity in outgoing SIP dialogs in the Outbound Manipulation table (using the column PrivacyRestrictionMode). The device identifies an incoming user as restricted if one of the following exists:

- From header user is 'anonymous'.
- P-Asserted-Identity and Privacy headers contain the value 'id'.

All restriction logic is done after the user number has been manipulated.

Host name (source and destination) manipulations are simply host name substitutions with the names defined for the source and destination IP Groups respectively (if any, in the IP Groups table).

Below is an example of a call flow and consequent SIP URI manipulations:

- **Incoming INVITE from LAN:**

```
INVITE sip:1000@10.2.2.3;user=phone;x=y;z=a SIP/2.0
Via: SIP/2.0/UDP 10.2.2.6;branch=z9hGLLLLLan
```

```
From:<sip:7000@10.2.2.6;user=phone;x=y;z=a>;tag=OlLAN;paramer1=abe
To: <sip:1000@10.2.2.3;user=phone>
Call-ID: USELLLAN@10.2.2.3
CSeq: 1 INVITE
Contact: <sip:7000@10.2.2.3>
Supported: em,100rel,timer,replaces
Allow: REGISTER,OPTIONS,INVITE,ACK,CANCEL,BYE,NOTIFY,PRACK
User-Agent: Sip Message Generator V1.0.0.5
Content-Type: application/sdp
Content-Length: 155
```

```
v=0
o=SMG 791285 795617 IN IP4 10.2.2.6
s=Phone-Call
c=IN IP4 10.2.2.6
t=0 0
m=audio 6000 RTP/AVP 8
a=rtpmap:8 pcma/8000
a=sendrecv
a=ptime:20
```

- **Outgoing INVITE to WAN:**

```
INVITE sip: 9721000@ITSP;user=phone;x=y;z=a SIP/2.0
Via: SIP/2.0/UDP 212.179.1.12;branch=z9hGWwan
From: <sip:97000@IP_PBX;user=phone;x=y;z=a>;tag=OWan;paramer1=abe
To: <sip: 9721000@ ITSP;user=phone>
Call-ID: USEVWWAN@212.179.1.12
CSeq: 38  INVITE
Contact: <sip:7000@212.179.1.12>
Supported: em,100rel,timer,replaces
Allow:
REGISTER,OPTIONS,INVITE,ACK,CANCEL,BYE,NOTIFY,PRACK,REFER
User-Agent: Sip Message Generator V1.0.0.5
Content-Type: application/sdp
Content-Length: 155
```

```
v=0
o=SMG 5  9  IN IP4 212.179.1.11
s=Phone-Call
c=IN IP4 212.179.1.11
t=0 0
m=audio 8000 RTP/AVP 8
a=rtpmap:8 pcma/8000
```

```
a=sendrecv
a=ptime:20
```

The SIP message manipulations in the example above (contributing to typical topology hiding) are as follows:

- Inbound source SIP URI user name from "7000" to "97000":

```
From:<sip:7000@10.2.2.6;user=phone;x=y;z=a>;tag=OILAN;paramer1=abe
```

to

```
From: <sip:97000@IP_PBX;user=phone;x=y;z=a>;tag=OWan;paramer1=abe
```

- Source IP Group name (i.e., SIP URI host name) from "10.2.2.6" to "IP_PBX":

```
From:<sip:7000@10.2.2.6;user=phone;x=y;z=a>;tag=OILAN;paramer1=abe
```

to

```
From: <sip:97000@IP_PBX;user=phone;x=y;z=a>;tag=OWan;paramer1=abe
```

- Inbound destination SIP URI user name from "1000" to 9721000":

```
INVITE sip:1000@10.2.2.3;user=phone;x=y;z=a SIP/2.0
```

```
To: <sip:1000@10.2.2.3;user=phone>
```

to

```
INVITE sip:9721000@ITSP;user=phone;x=y;z=a SIP/2.0
```

```
To: <sip:9721000@ITSP;user=phone>
```

- Destination IP Group name (SIP URI host name) from "10.2.2.3" to "ITSP":

```
INVITE sip:1000@10.2.2.3;user=phone;x=y;z=a SIP/2.0
```

```
To: <sip:1000@10.2.2.3;user=phone>
```

to

```
INVITE sip:9721000@ITSP;user=phone;x=y;z=a SIP/2.0
```

To: <sip:9721000@ITSP;user=phone>

Configuring IP-to-IP Inbound Manipulations

The Inbound Manipulations table lets you configure up to 1,250 IP-to-IP Inbound Manipulation rules. An Inbound Manipulation rule defines a manipulation sequence for the source or destination SIP URI user part of inbound SIP dialog requests. You can apply these manipulations to different SIP dialog message types (e.g., INVITE or REGISTER) and SIP headers as follows:

- Manipulated **destination URI user part** are done on the following SIP headers: Request-URI and To
- Manipulated **source URI user part** are done on the following SIP headers: From, P-Asserted-Identity (if exists), P-Preferred-Identity (if exists), and Remote-Party-ID (if exists)

Manipulated URI user part of the SIP From and Request-URI headers overwrite the user part of other headers.

Configuration of Inbound Manipulation rules includes two areas:

- **Match:** Defines the matching characteristics of an incoming SIP dialog (e.g., source host name).
- **Action:** Defines the action that is done if the incoming call matches the characteristics of the rule. In other words, the device manipulates the source or destination SIP URI user part of the SIP dialog (e.g., removes a user-defined number of characters from the left of the SIP URI user part).

Configure stricter classification rules higher up in the table than less strict rules to ensure the desired rule is used to manipulate the incoming dialog. *Strict* refers to the number of matching characteristics configured for the rule. For example, a rule configured with source host name and source IP Group as matching characteristics is stricter than a rule configured with only source host name. If the rule configured with only source host name appears higher up in the table, the device ("erroneously") uses the rule to manipulate incoming dialogs matching this source host name (even if they also match the rule appearing lower down in the table configured with the source IP Group as well).

To configure and apply an Inbound Manipulation rule, the rule must be associated with a Routing Policy. The Routing Policy associates the rule with an SRD(s). Therefore, the Routing Policy lets you configure manipulation rules for calls belonging to specific SRD(s). However, as multiple Routing Policies are relevant only for multi-tenant deployments (if needed), for most deployments, only a single Routing Policy is required. As the device provides a default Routing Policy ("Default_SBCRoutingPolicy"), when only one Routing Policy is required, the device automatically assigns the default Routing Policy to the routing rule. If you are implementing LDAP-based routing (with or without Call Setup Rules) and/or Least Cost Routing (LCR), you need to configure these settings for the Routing Policy (regardless of the number of Routing

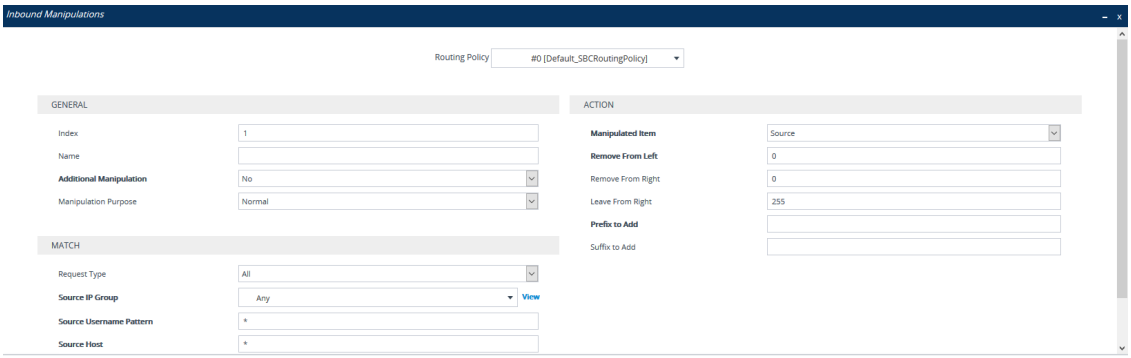
Policies employed). For more information on Routing Policies, see Configuring SBC Routing Policy Rules.

> The IP Groups table can be used to configure a host name that overwrites the received host name. This manipulation can be done for source and destination IP Groups (see Configuring IP Groups).

The following procedure describes how to configure Inbound Manipulation rules through the Web interface. You can also configure it through ini file [IPInboundManipulation] or CLI (`configure voip > sbc manipulation ip-inbound-manipulation`).

➢ To configure an Inbound Manipulation rule:

1. Open the Inbound Manipulations table (**Setup** menu > **Signaling & Media** tab > **SBC** folder > **Manipulation** > **Inbound Manipulations**).
2. Click **New**; the following dialog box appears:



3. Configure the Inbound Manipulation rule according to the parameters described in the table below.
4. Click **Apply**.

Table 26-1: Inbound Manipulations Table Parameter Descriptions

'Routing Policy' `routing-policy-name` [IPInboundManipulation_RoutingPolicyName]	Assigns an Routing Policy to the rule. The Routing Policy associates the rule with an SRD(s). The Routing Policy also defines default LCR settings as well as the LDAP servers if the routing rule is based on LDAP routing (and Call Setup Rules). If only one Routing Policy is configured in the Routing Policies table, the Routing Policy is automatically assigned. If multiple Routing Policies are configured, no value is assigned. To configure Routing Policies, see Configuring SBC Routing Policy Rules.

	Note: The parameter is mandatory.
General	
'Index' [IPInboundManipulation_Index]	Defines an index number for the new table record. **Note:** Each table row must be configured with a unique index.
'Name' `manipulation-name` [IPInboundManipulation_ManipulationName]	Defines an arbitrary name to easily identify the manipulation rule. The valid value is a string of up to 40 characters. By default, no value is defined.
'Additional Manipulation' `is-additional-manipulation` [IPInboundManipulation_IsAdditionalManipulation]	Determines whether additional SIP URI user part manipulation is done for the table entry rule listed directly above it. ■ [0] **No** = (Default) Regular manipulation rule (not done in addition to the rule above it). ■ [1] **Yes** = If the above row entry rule matched the call, consider this row entry as a match as well and perform the manipulation specified by this rule. **Note:** Additional manipulation can only be done on a different SIP URI, source or destination, to the rule configured in the row above as configured by the 'Manipulated URI' parameter (see below).
'Manipulation Purpose' `purpose` [IPInboundManipulation_ManipulationPurpose]	Defines the purpose of the manipulation. ■ [0] **Normal** = (Default) Inbound manipulations affect the routing input and source and/or destination number. ■ [1] **Routing input only** = Inbound manipulations affect the routing input only, retaining the original source and destination number. ■ [2] **Shared Line** = Used for the Shared-Line Appearance feature. This manipulation is for registration requests to change the destination number of the secondary extension numbers to the primary extension. For more information, see Configuring BroadSoft's Shared Phone Line Call Appearance for Survivability.

Match	
'Request Type' request-type [IPInboundManipulation_RequestType]	Defines the SIP request type to which the manipulation rule is applied. ■ [0] **All** = (Default) All SIP messages. ■ [1] **INVITE** = All SIP messages except REGISTER and SUBSCRIBE. ■ [2] **REGISTER** = Only REGISTER messages. ■ [3] **SUBSCRIBE** = Only SUBSCRIBE messages. ■ [4] **INVITE and REGISTER** = All SIP messages except SUBSCRIBE. ■ [5] **INVITE and SUBSCRIBE** = All SIP messages except REGISTER.
'Source IP Group' src-ip-group-name [IPInboundManipulation_SrcIpGroupName]	Defines the IP Group from where the incoming INVITE is received. The default is **Any** (i.e., any IP Group).
'Source Username Pattern' src-user-name-pattern [IPInboundManipulation_SrcUsernamePrefix]	Defines the source SIP URI user name (usually in the From header). The default is the asterisk (*) symbol (i.e., any source user name). You can use special pattern notations to denote the user part. For available notations, see Dialing Plan Notation for Routing and Manipulation.
'Source Host' src-host [IPInboundManipulation_SrcHost]	Defines the source SIP URI host name - full name (usually in the From header). The default is the asterisk (*) symbol (i.e., any host name).
'Destination Username Pattern' dst-user-name-pattern [IPInboundManipulation_DestUsernamePrefix]	Defines the destination SIP URI user name, typically located in the Request-URI and To headers. The default is the asterisk (*) symbol (i.e., any destination user name). You can use special pattern notations to denote the user part. For available notations, see Dialing Plan Notation for Routing and Manipulation.
'Destination Host' dst-host	Defines the destination SIP URI host name - full name, typically located in the Request URI and To headers.

[IPInboundManipulation_ DestHost]	The default is the asterisk (*) symbol (i.e., any destination host name).
Operation Rule - Action	
'Manipulated Item' `manipulated-uri` [IPInboundManipulation_ ManipulatedURI]	Determines whether the source or destination SIP URI user part is manipulated. ■ [0] **Source** = (Default) Manipulation is done on the source SIP URI user part. ■ [1] **Destination** = Manipulation is done on the destination SIP URI user part.
'Remove From Left' `remove-from-left` [IPInboundManipulation_ RemoveFromLeft]	Defines the number of digits to remove from the left of the user name prefix. For example, if you enter 3 and the user name is "john", the new user name is "n".
'Remove From Right' `remove-from-right` [IPInboundManipulation_ RemoveFromRight]	Defines the number of digits to remove from the right of the user name prefix. For example, if you enter 3 and the user name is "john", the new user name is "j". **Note:** If both 'Remove From Right' and 'Leave From Right' parameters are configured, the 'Remove From Right' setting is applied first.
'Leave From Right' `leave-from-right` [IPInboundManipulation_ LeaveFromRight]	Defines the number of characters that you want retained from the right of the user name. **Note:** If both 'Remove From Right' and 'Leave From Right' parameters are configured, the 'Remove From Right' setting is applied first.
'Prefix to Add' `prefix-to-add` [IPInboundManipulation_ Prefix2Add]	Defines the number or string that you want added to the front of the user name. For example, if you enter 'user' and the user name is "john", the new user name is "userjohn".
'Suffix to Add' `suffix-to-add` [IPInboundManipulation_ Suffix2Add]	Defines the number or string that you want added to the end of the user name. For example, if you enter '01' and the user name is "john", the new user name is "john01".

Configuring IP-to-IP Outbound Manipulations

The Outbound Manipulations table lets you configure up to 1,250 IP-to-IP Outbound Manipulation rules. An Outbound Manipulation rule defines a manipulation action for the SIP Request-URI user part (source or destination) or calling name of outbound SIP dialog requests. You can apply these manipulations to different SIP request types (e.g., INVITE) and SIP headers as follows:

- Manipulated **destination URI user part** are done on the following SIP headers: Request URI and To
- Manipulated **source URI user part** are done on the following SIP headers: From, P-Asserted (if exists), P-Preferred (if exists), and Remote-Party-ID (if exists)

> Manipulated URI user part of the SIP From and Request-URI headers overwrite the user part of other headers.

Configuration of Outbound Manipulation rules includes two areas:

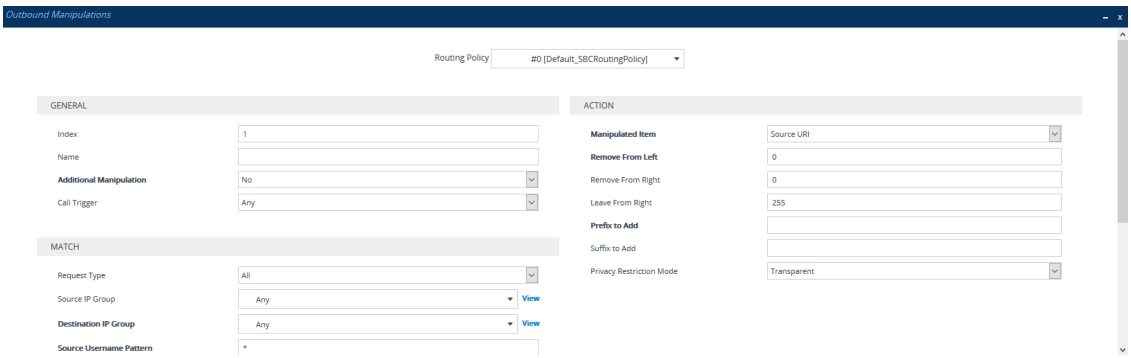
- **Match:** Defines the matching characteristics of an incoming SIP dialog (e.g., source host name). As the device performs outbound manipulations only after the routing process, destination IP Groups can also be used as matching characteristics.
- **Action:** Defines the action that is done if the incoming call matches the characteristics of the rule. In other words, the device manipulates the source or destination SIP URI user part or calling name of the SIP dialog (e.g., removes a user-defined number of characters from the left of the SIP URI user part).

> - Configure stricter classification rules higher up in the table than less strict rules to ensure the desired rule is used to manipulate the outbound dialog. *Strict* refers to the number of matching characteristics configured for the rule. For example, a rule configured with source host name and source IP Group as matching characteristics is stricter than a rule configured with only source host name. If the rule configured with only source host name appears higher up in the table, the device ("erroneously") uses the rule to manipulate outbound dialogs matching this source host name (even if they also match the rule appearing lower down in the table configured with the source IP Group as well).
> - SIP URI host name (source and destination) manipulations can also be configured in the IP Groups table (see Configuring IP Groups). These manipulations are simply host name substitutions with the names configured for the source and destination IP Groups, respectively.

The following procedure describes how to configure Outbound Manipulations rules through the Web interface. You can also configure it through ini file [IPOutboundManipulation] or CLI (`configure voip > sbc manipulation ip-outbound-manipulation`).

➢ To configure Outbound Manipulation rules:

1. Open the Outbound Manipulations table (**Setup** menu > **Signaling & Media** tab > **SBC** folder > **Manipulation** > **Outbound Manipulations**).
2. Click **New**; the following dialog box appears:



3. Configure an Outbound Manipulation rule according to the parameters described in the table below.
4. Click **Apply**.

Table 26-2: Outbound Manipulations Table Parameter Description

Parameter	Description
'Routing Policy' routing-policy-name [IPOutboundManipulation_RoutingPolicyName]	Assigns a Routing Policy to the rule. The Routing Policy associates the rule with an SRD(s). The Routing Policy also defines default LCR settings as well as the LDAP servers if the routing rule is based on LDAP routing (and Call Setup Rules). If only one Routing Policy is configured in the Routing Policies table, the Routing Policy is automatically assigned. If multiple Routing Policies are configured, no value is assigned. To configure Routing Policies, see Configuring SBC Routing Policy Rules. **Note:** The parameter is mandatory.
General	
'Index' [IPOutboundManipulation_Index]	Defines an index number for the new table row. **Note:** Each row must be configured with a unique index.
'Name' manipulation-name [IPOutboundManipulation_ManipulationName]	Defines a descriptive name, which is used when associating the row in other tables. The valid value is a string of up to 40 characters. By default, no value is defined.

Parameter	Description
'Additional Manipulation' `is-additional-manipulation` [IPOutboundManipulation_ IsAdditionalManipulation]	Determines whether additional manipulation is done for the table entry rule listed directly above it. ■ [0] **No** = (Default) Regular manipulation rule - not done in addition to the rule above it. ■ [1] **Yes** = If the previous table row entry rule matched the call, consider this row entry as a match as well and perform the manipulation specified by this rule. **Note:** Additional manipulation can only be done on a different item (source URI, destination URI, or calling name) to the rule configured in the row above (configured by the 'Manipulated URI' parameter).
'Call Trigger' `trigger` [IPOutboundManipulation_Trigger]	Defines the reason (i.e., trigger) for the re-routing of the SIP request. ■ [0] **Any** = (Default) Re-routed for all scenarios (re-routes and non-re-routes). ■ [1] **3xx** = Re-routed if it triggered as a result of a SIP 3xx response. ■ [2] **REFER** = Re-routed if it triggered as a result of a REFER request. ■ [3] **3xx or REFER** = Applies to options [1] and [2]. ■ [4] **Initial only** = Regular requests that the device forwards to a destination. In other words, re-routing of requests triggered by the receipt of REFER or 3xx does not apply.
Match	
'Request Type' `request-type` [IPOutboundManipulation_RequestType]	Defines the SIP request type to which the manipulation rule is applied. ■ [0] **All** = (Default) all SIP messages. ■ [1] **INVITE** = All SIP messages except REGISTER and SUBSCRIBE. ■ [2] **REGISTER** = Only SIP REGISTER messages. ■ [3] **SUBSCRIBE** = Only SIP SUBSCRIBE messages. ■ [4] **INVITE and REGISTER** = All SIP messages except SUBSCRIBE. ■ [5] **INVITE and SUBSCRIBE** = All SIP messages except

Parameter	Description
	REGISTER.
'Source IP Group' src-ip-group-name [IPOutboundManipulation_SrcIPGroupName]	Defines the IP Group from where the INVITE is received. The default value is **Any** (i.e., any IP Group).
'Destination IP Group' dst-ip-group-name [IPOutboundManipulation_DestIPGroupName]	Defines the IP Group to where the INVITE is to be sent. The default value is **Any** (i.e., any IP Group).
'Source Username Pattern' src-user-name-pattern [IPOutboundManipulation_SrcUsernamePrefix]	Defines the source SIP URI user name (typically used in the SIP From header). You can use special patterns (notations) to denote the user part. For example, if you want to match this rule to user parts whose last four digits (i.e., suffix) are 4 followed by any three digits (e.g., 4008), then configure this parameter to "(4xxx)". For available patterns, see Dialing Plan Notation for Routing and Manipulation. The valid value is a string of up to 60 characters. The default value is the asterisk (*) symbol, meaning any source user part. **Note:** If you need to manipulate calls of many different source URI user parts, you can use tags (see 'Source Tags' parameter below) instead of this parameter.
'Source Host' src-host [IPOutboundManipulation_SrcHost]	Defines the source SIP URI host name - full name, typically in the From header. The default value is the asterisk (*) symbol (i.e., any source host name).
'Source Tags' src-tags [IPOutboundManipulation_SrcTags]	Assigns a prefix tag to denote source URI user names corresponding to the tag configured in the associated Dial Plan. The valid value is a string of up to 70 characters. The tag is case insensitive. By default, no value is defined. You can configure the parameter with up to five tags, where each tag is separated by a semicolon (;). However, you can configure only up to four tags containing a name and value (e.g., Country=Ireland), and **one** tag containing a value only (e.g., Ireland). If you are configuring multiple tags in the name=value format, the names

Parameter	Description
	of each tag must be unique (e.g., **Country**=Ireland;**Land**=Scotland). The following example configures the maximum number of tags (i.e., four name=value tags and one value-only tag): Country=Ireland;Country2=Scotland;Country3=RSA;Country4=Canada;USA. To configure prefix tags, see Configuring Dial Plans. **Note:** ■ Make sure that you assign the Dial Plan in which you have configured the prefix tag, to the related IP Group or SRD. ■ Instead of using tags and configuring the parameter, you can use the 'Source Username Pattern' parameter to specify a specific URI source user or all source users.
'Destination Username Pattern' `dst-user-name-pattern` [IPOutboundManipulation_ DestUsernamePrefix]	Defines the destination SIP URI user part (typically located in the Request-URI and To headers). You can use special patterns (notations) to denote the user part. For example, if you want to match this rule to user parts whose last four digits (i.e., suffix) are 4 followed by any three digits (e.g., 4008), then configure this parameter to "(4xxx)". For available patterns, see Dialing Plan Notation for Routing and Manipulation. The valid value is a string of up to 60 characters. The default value is the asterisk (*) symbol, meaning any destination user part. **Note:** If you need to manipulate calls of many different destination URI user names, you can use tags (see 'Destination Tags' parameter below) instead of this parameter.
'Destination Host' `dst-host` [IPOutboundManipulation_DestHost]	Defines the destination SIP URI host name - full name, typically located in the Request-URI and To headers. The default value is the asterisk (*) symbol (i.e., any destination host name).
'Destination Tags' `dest-tags` [IPOutboundManipulation_DestTags]	Assigns a prefix tag to denote destination URI user names corresponding to the tag configured in the associated Dial Plan. The valid value is a string of up to 70 characters. The tag is case insensitive. By default, no value is defined. You can configure the parameter with up to five tags, where each tag is separated by a semicolon (;). However, you can configure only up to four tags containing a name and value (e.g., Country=Ireland), and

Parameter	Description
	one tag containing a value only (e.g., Ireland). If you are configuring multiple tags in the name=value format, the names of each tag must be unique (e.g., **Country**=Ireland;**Land**=Scotland). The following example configures the maximum number of tags (i.e., four name=value tags and one value-only tag): Country=Ireland;Country2=Scotland;Country3=RSA;Country4=Canada;USA. To configure prefix tags, see Configuring Dial Plans. **Note:** ■ Make sure that you assign the Dial Plan in which you have configured the prefix tag, to the related IP Group or SRD. ■ Instead of using tags and configuring the parameter, you can use the 'Destination Username Pattern' parameter to specify a specific URI destination user or all destinations users.
'Calling Name Pattern' `calling-name-pattern` [IPOutboundManipulation_CallingNamePrefix]	Defines the calling name (caller ID). The calling name appears in the SIP From header. The valid value is a string of up to 37 characters. By default, no calling name is defined. You can use special patterns (notations) to denote the calling name. For available patterns, see Dialing Plan Notation for Routing and Manipulation.
'Message Condition' `message-condition-name` [IPOutboundManipulation_ MessageConditionName]	Assigns a Message Condition rule as a matching characteristic. Message Condition rules define required SIP message formats. To configure Message Condition rules, see Configuring Message Condition Rules.
'ReRoute IP Group' `re-route-ip-group-name` [IPOutboundManipulation_ ReRouteIPGroupName]	Defines the IP Group that initiated (sent) the SIP redirect response (e.g., 3xx) or REFER message. The parameter is typically used for re-routing requests (e.g., INVITEs) when interworking is required for SIP 3xx redirect responses or REFER messages. The default is **Any** (i.e., any IP Group). **Note:** ■ The parameter functions together with the 'Call Trigger' parameter (see below).

Parameter	Description
	■ For more information on interworking of SIP 3xx redirect responses or REFER messages, see Interworking SIP 3xx Redirect Responses and Interworking SIP REFER Messages, respectively.
Action	
'Manipulated Item' `manipulated-uri` [IPOutboundManipulation_ IsAdditionalManipulation]	Defines the element in the SIP message that you want manipulated. ■ [0] **Source URI** = (Default) Manipulates the source SIP Request-URI user part. ■ [1] **Destination URI** = Manipulates the destination SIP Request-URI user part. ■ [2] **Calling Name** = Manipulates the calling name in the SIP message.
'Remove From Left' `remove-from-left` [IPOutboundManipulation_RemoveFromLeft]	Defines the number of digits to remove from the left of the manipulated item prefix. For example, if you enter 3 and the user name is "john", the new user name is "n".
'Remove From Right' `remove-from-right` [IPOutboundManipulation_RemoveFromRight]	Defines the number of digits to remove from the right of the manipulated item prefix. For example, if you enter 3 and the user name is "john", the new user name is "j".
'Leave From Right' `leave-from-right` [IPOutboundManipulation_LeaveFromRight]	Defines the number of digits to keep from the right of the manipulated item.
'Prefix to Add' `prefix-to-add` [IPOutboundManipulation_Prefix2Add]	Defines the number or string to add in the front of the manipulated item. For example, if you enter 'user' and the user name is "john", the new user name is "userjohn". If you set the 'Manipulated Item' parameter to **Source URI** or **Destination URI**, you can configure the parameter to a string of up 49 characters. If you set the 'Manipulated Item' parameter to **Calling Name**, you can configure the parameter to a string of up to 36 characters.

Parameter	Description
'Suffix to Add' suffix-to-add [IPOutboundManipulation_Suffix2Add]	Defines the number or string to add at the end of the manipulated item. For example, if you enter '01' and the user name is "john", the new user name is "john01". If you set the 'Manipulated Item' parameter to **Source URI** or **Destination URI**, you can configure the parameter to a string of up 49 characters. If you set the 'Manipulated Item' parameter to **Calling Name**, you can configure the parameter to a string of up to 36 characters.
'Privacy Restriction Mode' privacy-restriction-mode [IPOutboundManipulation_PrivacyRestrictionMode]	Defines user privacy handling (i.e., restricting source user identity in outgoing SIP dialogs). ■ [0] **Transparent** = (Default) No intervention in SIP privacy. ■ [1] **Don't change privacy** = The user identity remains the same as in the incoming SIP dialog. If a restricted number exists, the restricted presentation is normalized as follows: ✓ From URL header: "anonymous@anonymous.invalid" ✓ If a P-Asserted-Identity header exists (either in the incoming SIP dialog or added by the device), a Privacy header is added with the value "id". ■ [2] **Restrict** = The user identity is restricted. The restriction is as follows: ✓ From URL header: "anonymous@anonymous.invalid" ✓ If a P-Asserted-Identity header exists (either in the incoming SIP dialog or added by the device), a Privacy header is added with the value "id". ■ [3] **Remove Restriction** = The device attempts to reveal the user identity by setting user values in the From header and removing the privacy "id" value if the Privacy header exists. If the From header user is "anonymous", the value is taken from the P-Preferred-Identity, P-Asserted-Identity, or Remote-Party-ID header (if exists). **Note:** ■ Restriction is done only after user number manipulation, if any. ■ The device identifies an incoming user as restricted if one of the following exists: ✓ From header user is "anonymous".

Parameter	Description
	✓ P-Asserted-Identity and Privacy headers contain the value "id".

Using the Proprietary SIP X-AC-Action Header

You can use AudioCodes proprietary SIP header, X-AC-Action in Message Manipulation rules to trigger certain actions. These actions can be used to support, for example, interworking of SIP-I and SIP endpoints for the ISUP SPIROU variant (see Enabling Interworking of SIP and SIP-I Endpoints).

The following actions are supported by the X-AC-Action header:

- **To disconnect a call (optionally, after a user-defined time):**

```
X-AC-Action: 'disconnect'
```

```
X-AC-Action: 'disconnect;delay=<time in ms>'
```

- **To resume a previously suspended call:**

```
X-AC-Action: 'abort-disconnect'
```

- **To automatically reply to a message without forwarding the response to the other side:**

```
X-AC-Action: 'reply'
```

- **To automatically reply to a message with a specific SIP response without forwarding the response to the other side:**

```
X-AC-Action: 'reply;response=<response code, e.g., 200>'
```

- **To override the device's handling of SIP REFER messages, which is configured by the 'Remote REFER Mode' [IpProfile_SBCRemoteReferBehavior] parameter.** The X-AC-Action header can be added to the incoming SIP REFER request using Message Manipulation rules. This is useful if you don't want the settings of this parameter to apply to all calls that are associated with the IP Profile. For example, if you configure the 'Remote REFER Mode' parameter to **Handle Locally**, all incoming SIP REFER requests associated with the IP Profile are terminated at the device. However, you can configure a Message Manipulation rule with the proprietary header to override this parameter setting and allow the device to forward the REFER requests as is for calls with a specific URI, for example. You can configure Message Manipulation rules to add this X-AC-Action header for REFER handling, with one of the following values:

- To allow the device to forward the REFER as is, regardless of the 'Remote REFER Mode' parameter settings:

```
X-AC-Action: 'use-config;refer-behavior=regular'
```

- To allow the device to handle (terminate) the REFER request regardless of the 'Remote REFER Mode' parameter settings:

```
X-AC-Action: 'use-config;refer-behavior= handle-locally'
```

- **To switch to a different IP Profile for the call (re-INVITE only), as defined in the IP Group:**

```
X-AC-Action: 'switch-profile;profile-name=<IP Profile Name>'
```

```
X-AC-Action: 'switch-profile;profile-name=<IP Profile Name>;reason=<PoorInVoiceQuality or PoorInVoiceQualityFailure>'
```

If the IP Profile name contains one or more spaces (e.g., "ITSP NET"), enclose the name in double quotation marks, for example:

```
X-AC-Action: 'switch-profile;profile-name="ITSP NET"'
```

For example, to use the X-AC-Action header to switch IP Profiles from "ITSP-Profile-1" to "ITSP-Profile-2" during a call for an IP Group (e.g., IP PBX) if the negotiated media port changes to 7550, perform the following configuration:

1. In the IP Profiles table, configure two IP Profiles ("ITSP-Profile-1" and "ITSP-Profile-2").
2. In the IP Groups table, assign the main IP Profile ("ITSP-Profile-1") to the IP Group using the 'IP Profile' parameter.
3. In the Message Manipulations table (see Configuring SIP Message Manipulation), configure the following manipulation rule:
   - Manipulation Set ID: **1**
   - Message Type: **reinvite.request**
   - Condition: **body.sdp regex (.*)(m=audio 7550 RTP/AVP)(.*)**
   - Action Subject: **header.X-AC-Action**
   - Action Type: **Add**
   - Action Value: **'switch-profile;profile-name=ITSP-Profile-2'**
4. In the IP Groups table, assign the Message Manipulation rule to the IP Group, using the 'Inbound Message Manipulation Set' parameter.

In the above example, if the device receives from the IP Group a re-INVITE message whose media port value is 7550, the device adds the SIP header "X-AC-Action: switch-profile;profile-name=ITSP- Profile- 2" to the incoming re- INVITE message. As a result of receiving this manipulated message, the device starts using IP Profile "ITSP-Profile-2" instead of "ITSP-Profile-1", for the IP Group.

27 Configuring Malicious Signatures

The Malicious Signature table lets you configure up to 20 Malicious Signature patterns. Malicious Signatures are signature patterns that identify SIP user agents (UA) who perform malicious attacks on SIP servers by SIP scanning. Malicious Signatures allow you to protect SBC calls handled by the device from such malicious activities, thereby increasing your SIP security. The Malicious Signature patterns identify specific scanning tools used by attackers to search for SIP servers in the network. The feature identifies and protects against SIP (Layer 5) threats by examining new inbound SIP dialog messages. Once the device identifies an attack based on the configured malicious signature pattern, it marks the SIP message as invalid and discards it or alternatively, rejects it with a SIP response (by default, 400), configured in the Message Policies table. Protection applies only to new dialogs (e.g., INVITE and REGISTER messages) and unauthenticated dialogs.

Malicious signatures can also be used with the Intrusion Detection System (IDS) feature (see Configuring IDS Policies). You can configure an IDS Policy that is activated if the device detects a malicious signature (when the 'Reason' parameter is configured to **Dialog establishment failure**).

Malicious signature patterns are typically based on the value of SIP User-Agent headers, which attackers use as their identification string (e.g., "User-Agent: VaxSIPUserAgent"). However, you can configure signature patterns based on any SIP header. To configure signature patterns, use the same syntax as that used for configuring Conditions in the Message Manipulations table (see Configuring SIP Message Manipulation). Below are configured signature patterns based on the User-Agent header:

- Malicious signature for the VaxSIPUserAgent malicious UA:

```
header.user-agent prefix 'VaxSIPUserAgent'
```

- Malicious signature for the scanning tool "sip-scan":

```
header.user-agent prefix 'sip-scan'
```

By default, the table provides preconfigured malicious signatures of known, common attackers.

- Malicious Signatures do not apply to the following:
  - ✓ Calls from IP Groups where Classification is by Proxy Set.
  - ✓ In-dialog SIP sessions (e.g., refresh REGISTER requests and re-INVITEs).
  - ✓ Calls from users that are registered with the device.
- If you delete all the entries in the table, when you next reset the device, the table is populated again with all the default signatures.

You can export / import Malicious Signatures in CSV file format to / from a remote server through HTTP, HTTPS, or TFTP. To do this, use the following CLI commands:

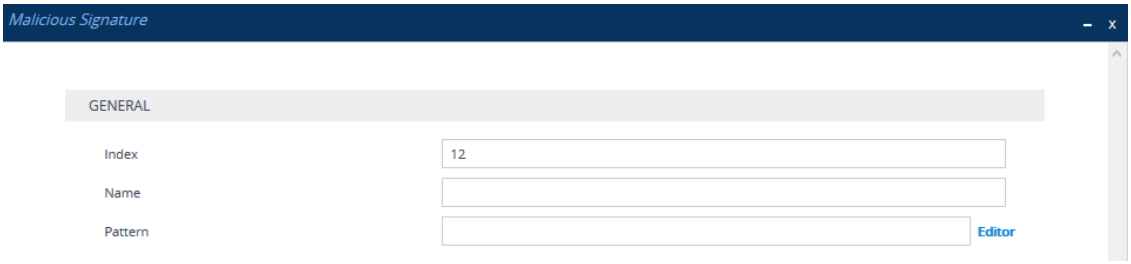
```
(config-voip)# sbc malicious-signature-database <export-csv-to | import-csv-from>
<URL>
```

To apply malicious signatures to calls, you need to enable the use of malicious signatures for a Message Policy and then assign the Message Policy to the SIP Interface associated with the calls (i.e., IP Group). To configure Message Policies, see Configuring SIP Message Policy Rules.

The following procedure describes how to configure Malicious Signatures through the Web interface. You can also configure it through ini file [MaliciousSignatureDB] or CLI (`configure voip > sbc malicious-signature-database`).

➢ To configure a Malicious Signature:

1. Open the Malicious Signature table (**Setup** menu > **Signaling & Media** tab > **SBC** folder > **Malicious Signature**).
2. Click **New**; the following dialog box appears:



3. Configure a Malicious Signature according to the parameters described in the table below.
4. Click **Apply**.

Table 27-1: Malicious Signature Table Parameter Descriptions

Parameter	Description
'Index' [ConditionTable_Index]	Defines an index number for the new table row. **Note:** Each row must be configured with a unique index.
'Name' `name` [MaliciousSignatureDB_Name]	Defines a descriptive name, which is used when associating the row in other tables. The valid value is a string of up to 30 characters. **Note:** Each row must be configured with a unique name.
'Pattern' `pattern` [MaliciousSignatureDB_Pattern]	Defines the signature pattern. The valid value is a string of up to 60 characters. You can use the built-in syntax editor to help you configure the field. Click the **Editor** button located alongside the field to open the Editor, and then simply follow the on-screen instructions. **Note:** The parameter is mandatory.

28 Advanced SBC Features

This section describes configuration of advanced SBC features.

Configuring Call Preemption for SBC Emergency Calls

The device supports emergency call preemption for SBC calls by prioritizing emergency calls over regular calls. If the device receives an incoming emergency call when there are unavailable resources to process the call, the device preempts one of the regular calls to free up resources for sending the emergency call to its' destination (i.e., emergency service provider), instead of rejecting it. The device may preempt more than one active call in order to provide sufficient resources for processing the emergency call. Available resources depends on the number of INVITE messages currently processed by the device.

If the device preempts a call, it disconnects the call as follows:

- If the call is being setup (not yet established), it sends a SIP 488 response to the incoming leg and a SIP CANCEL message to the outgoing leg.
- If the call is already established, it sends a SIP BYE message to each leg. The device includes in the SIP BYE message, the Reason header describing the cause as "preemption".

Once the device terminates the regular call, it immediately sends the INVITE message of the emergency call to its' destination without waiting for any response from the remote sides (e.g., 200 OK after BYE). If the device is unable to preempt a call for the emergency call, it rejects the emergency call with a SIP 503 "Emergency Call Failed" (instead of "Service Unavailable") response.

For the device to identify incoming calls as emergency calls, you need to configure a Message Condition rule in the Message Conditions table. Below are examples of Message Condition rules for identifying emergency calls:

Table 28-1: Examples of Message Condition Rules for Emergency Calls

Index	Name	Condition
0	Emergency1 - RP header	header.resource-priority contains 'emergency'
1	Emergency2 - RP header	header.resource-priority contains 'esnet'
2	Emergency1 - user with providers address	header.to.url.user=='911'
3	Emergency2 - user with providers address	header.to.url.user=='100' ||header.to.url.user=='101'||header.to.url.user=='102'

Index	Name	Condition
4	Emergency3 - user with providers address	header.request.uri contains 'urn:service:sos'

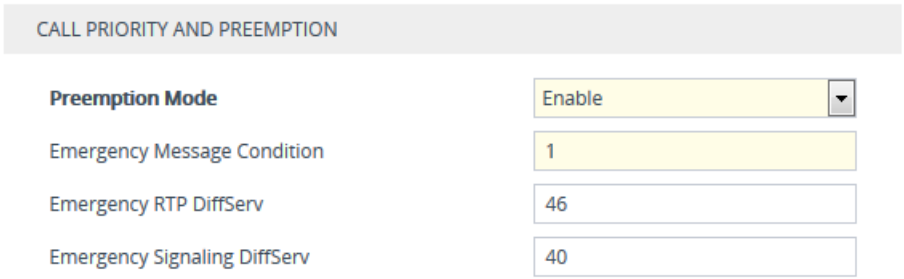
- Indices 0 and 1: SIP Resource-Priority header contains a string indicating an emergency call.
- Indices 2 to 4: Destination user-part contains the emergency provider's address.

The device applies the Message Condition rule only after call classification (but, before inbound manipulation).

The device does not preempt established emergency calls.

➢ To configure SBC emergency call preemption:

1. In the Message Conditions table (see Configuring Message Condition Rules), configure a Message Condition rule to identify incoming emergency calls. See above for examples.
2. Open the SBC General Settings page (**Setup** menu > **Signaling & Media** tab > **SIP Definitions** folder > **Priority and Emergency**), and then scroll down to the Call Priority and Preemption group:



3. From the 'Preemption Mode' drop-down list (SBCPreemptionMode), select **Enable** to enable call preemption.
4. In the 'Emergency Message Condition' field, enter the row index of the Message Condition rule that you configured in Step 1.
5. (Optional) Assign DiffServ levels (markings) to packets belonging to emergency calls:
   - In the 'Emergency RTP DiffServ' field (SBCEmergencyRTPDiffServ), enter the QoS level for RTP packets.
   - In the 'Emergency Signaling DiffServ' field (SBCEmergencySignalingDiffServ), enter the QoS level for SIP signaling packets.
6. Click **Apply**.

The call preemption feature uses only total licensed SBC signaling (SIP) resources and/or the Call Admission Control feature (see Configuring Call Admission Control on page 696), and not the number of configured available media (RTP) ports for determining an out-of-resources scenario. Therefore, it's highly recommended that you configure the number of media session legs in the Media Realm table with at least twice the number of SBC signaling resources (see Configuring Media Realms on page 410).

Configuring Message Session Relay Protocol

The device supports Message Session Relay Protocol (MSRP), which is a text-based protocol for exchanging a series of related instant messages (IM) across an IP network (TCP or TLS only) in the context of a session. The protocol can also be used to transfer large files or images, or share remote desktops or whiteboards. MSRP is typically required for Next Generation 911 (NG911) services, allowing 911 callers to not only access 911 services through voice calls, but also through text messages with Public Safety Answering Points (PSAPs). The device's MSRP support is in accordance with RFC 4975 (The Message Session Relay Protocol (MSRP)) and RFC 6135 (An Alternative Connection Model for the Message Session Relay Protocol (MSRP)). The device also supports secure MSRP sessions (MSRPS), using TLS certificates (TLS Context).

The device establishes MSRP sessions using the SDP offer/answer negotiation model over SIP. The MSRP session starts with a SIP INVITE and ends with a SIP BYE message. As a B2BUA, the device interoperates between the MSRP endpoints, terminating the incoming MSRP message on the inbound leg and then generating a new MSRP message on the outbound leg. Before sending the INVITE, the device manipulates the SDP body (e.g., 'a=path', 'c=', 'm=', 'a=setup' and 'a=fingerprint' lines). The device can perform optional message manipulation and other translations such as resolving NAT traversal when the endpoints or device are located behind NAT.

An example of an SDP body with the fields for MSRP negotiation in the INVITE message is shown below:

```
INVITE sip:alice@atlanta.example.com SIP/2.0
To: <sip:bob@biloxi.example.com>
From: <sip:alice@atlanta.example.com>;tag=786
Call-ID: 3413an89KU
Content-Type: application/sdp

c=IN IP4 atlanta.example.com
m=message 7654 TCP/MSRP *
a=accept-types:text/plain
a=path:msrp://atlanta.example.com:7654/jshA7weztas;tcp
a=setup:active
```

Where,

- 'c=' line ignores IP address and port
- 'm=' line indicates an MSRP message
- 'a=accept-types:' line lists allowed content types
- 'a=path:msrp:' line indicates the URI to where the messages are to be sent
- 'a=setup' line indicates the MSRP role (active UA initiates connection; passive UA listens on port)

If secured MSRP (MSRPS) is required (i.e., incoming SDP contains 'm=' line with 'TCP/TLS/MSRP' value, 'a=path' with 'msrps', and 'a=fingerprint'), during MSRP session establishment, the device enforces the validity of the fingerprint from the TLS handshake (public key) with the fingerprint in the received SDP. When the device establishes a secured MSRP session, the offered fingerprint is obtained from the TLS Context, which is assigned to the IP Profile of the endpoint.

The device handles MSRP sessions as follows:

1. When the device receives an INVITE message with the MSRP offer, it initiates an SDP offer to the destination endpoint on the outgoing leg. Before sending the INVITE message, the device does the following:
   - Chooses an unused MSRP listening port (SDP 'm=' line) from the TCP media port range configured for the associated Media Realm.
   - Uses the IP address (SDP 'c=' line) of the associated IP Interface (SIP Interface).
   - Sets the 'a=setup' line to the configured preferred MSRP role of the device.
2. When the device receives the MSRP answer from the destination endpoint, it sends an SDP answer to the dialog-initiating endpoint. Before sending the INVITE message, the device does the following:
   - If the device has chosen a TCP server role, it selects an unused listening port from the TCP media port range which is capable of accepting TCP connections. This port number is included in the media line of the SDP.
   - The device includes the IP address of the associated IP Interface in the 'c=' line of the SDP.
   - Sets the 'a=setup' line to the device's negotiated role.

Once SDP negotiation between the UAs is complete and the MSRP session is being established, the device initiates a TCP/TLS connection (or waits to be initiated) on each leg, depending on SDP negotiation. Once a TCP/TLS connection is established, the endpoints can start sending MSRP messages using MSRP SEND requests, as shown in the following example:

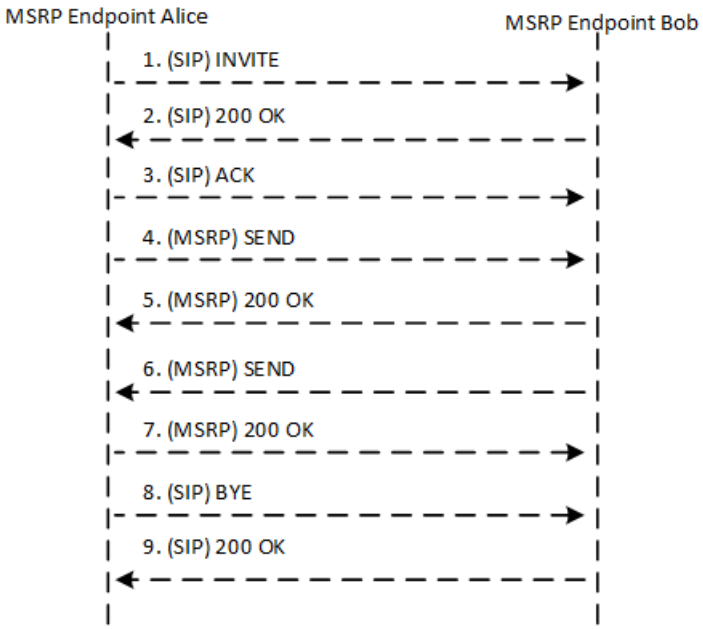
```
MSRP a786hjs2 SEND
To-Path: msrp://biloxi.example.com:12763/kjhd37s2s20w2a;tcp
From-Path: msrp://atlanta.example.com:7654/jshA7weztas;tcp
Message-ID: 87652491
Byte-Range: 1-25/25
Content-Type: text/plain
```

```
Hey Bob, are you there?
-------a786hjs2$
```

The MSRP payload or content (i.e. the actual message) follows the Content-Type header. Finally, the SEND request is closed with an end-line of seven hyphens, the Transaction ID, and one of the following symbols:

- **$:** The request contains the final part of the message (end).
- **+:** The request does not contain the final part of the message ($), but is only part of a series of messages.
- **#:** The sender is aborting an incomplete message and intends to send no further chunks in that message (message should be discarded).

An example of a basic MSRP flow is shown below:



- For device's operating in HA mode, MSRP sessions are dropped during an HA switchover. MSRP endpoints are expected to re-initiate failed sessions.
- MSRP is not supported with other media types (i.e., voice) in the same SDP session.
- The Call Admission Control (CAC) mechanism handles MSRP sessions as regular media sessions (i.e., they are not calculated and monitored separately from regular media calls).
- CDRs generated by the device for MSRP calls include the value "msrp" in the Media List CDR field.

The following procedure provides the basic steps for configuring MSRP.

➢ **To configure MSRP:**

1. Open the IP Profiles table (see Configuring IP Profiles on page 519), and then configure the MSRP endpoint's IP Profile with the following (in addition to other IP Profile settings that may require):
   - From the 'SBC Media Security Mode' drop-down list, select the transport protocol for the outgoing leg - **Secured** or **Both** for MSRPS (example below); **Not Secured** for MSRP.

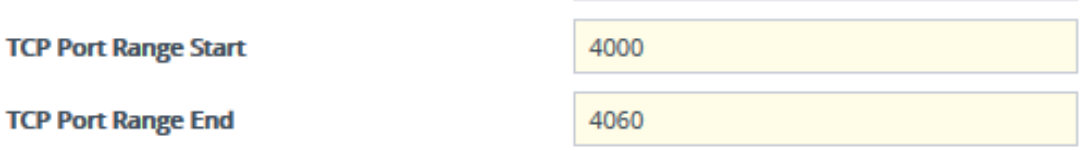
   - From the 'MSRP Offer Setup Role' drop-down list, select the MSRP role mode in SDP negotiations ('a=setup' line). The device's role is according to the response: If 'a=setup passive', it's the "active" role; if 'a=setup active', it's the "passive" role. If no 'a=setup' in the response, it's the "active" role.

   - In the 'Data DiffServ' field, configure the DiffServ value of MSRP traffic ('m=message').

   - From the 'MSRP re-INVITE/UPDATE' drop-down list, select if the destination MSRP endpoint supports the receipt of re-INVITE requests and UPDATE messages.

   - From the 'MSRP Empty Message Format' drop-down list, select if the device must add a Content-Type header to empty MSRP messages that are used to initiate the connection.

2. Open the Media Realms table (see Configuring Media Realms on page 410), and then configure the MSRP endpoint's Media Realm with MSRP ports in the 'TCP Port Range Start' and 'TCP Port Range End' fields. The port number is used in the SDP's 'a=path' line.



3. Open the IP Groups table (see Configuring IP Groups on page 451), and then for secured MSRP (MSRPS), assign a TLS Context (certificate) to the endpoint's IP Group, using the 'Media TLS Context' parameter.

4. For NAT traversal of MSRP sessions when the device is located behind NAT, use the NAT Translation table (see Configuring NAT Translation per IP Interface on page 139). The target IP address:port (public) is used in the SDP's 'a=path' line.

Emergency Call Routing using LDAP to Obtain ELIN

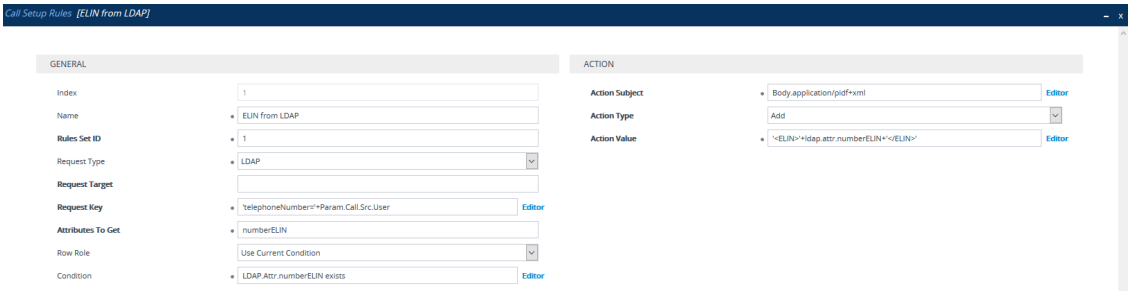
The device can route emergency calls (e.g., 911) for INVITE messages that are received without an ELIN number. This is in contrast to when the device is deployed in a Microsoft Teams / Skype for Business environment, whereby the received INVITE messages contain ELIN numbers. For a detailed explanation on ELIN numbers and handling of emergency calls by emergency server providers, see E9-1-1 Support for Microsoft Teams and Skype for Business on page 369.

To obtain an ELIN number for emergency calls received without ELINs, you can configure the device to query an LDAP server for the 911 caller's ELIN number. The device adds the resultant ELIN number and a Content-Type header for the PIDF XML message body to the outgoing INVITE message, for example:

```
Content-Type: application/pidf+xml
        <ELIN>1234567890</ELIN>
```

➢ To enable emergency call routing using LDAP to obtain ELIN:

1. Configure a Call Setup rule in the Call Setup Rules table (see Configuring Call Setup Rules). The following example shows a Call Setup rule that queries an Active Directory (AD) server for the attribute "telephoneNumber" whose value is the E9-1-1 caller's number (source), and then retrieves the user's ELIN number from the attribute "numberELIN":



2. Enable the E9-1-1 feature, by configuring the 'PSAP Mode' parameter to **PSAP Server** in the IP Groups table for the IP Group of the PSAP server (see Enabling the E9-1-1 Feature).

3. Configure routing rules in the IP-to-IP Routing table for routing between the emergency callers' IP Group and the PSAP server's IP Group. The only special configuration required for the routing rule from emergency callers to the PSAP server:
   - Configure the emergency number (e.g., 911) in the 'Destination Username Pattern' field.
   - Assign the Call Setup rule that you configured for obtaining the ELIN number from the AD (see Step 1) in the 'Call Setup Rules Set ID' field (see Configuring SBC IP-to-IP Routing Rule for E9-1-1).

Configuring Dual Registration for SIP Entity

Some SIP entities (e.g., IP Phones) are setup to register with two registrar/proxy servers (primary and secondary). The reason for this is to provide call redundancy for the SIP entity in case one of the proxy servers fail. When the SIP entity registers with the proxy servers, it sends two identical REGISTER messages - one to the primary proxy and one to the secondary proxy. When the device is located between the SIP entity and the two proxy servers, it needs to differentiate between these two REGISTER messages even though they are identical. This is crucial to ensure that the device forwards the two registrations to the proxy servers and that the device performs correct call routing between the SIP entity and the two proxy servers.

To differentiate between these REGISTER messages, a unique SIP Interface needs to be used for each REGISTER message. Each REGISTER message is registered in the registration database using a unique "ac-int=<value>" string identifying the SIP Interface for the Contact user. In addition, for SIP requests (e.g., INVITE) from the proxy servers, the device needs to search its registration database for the contact user so that it can forward it to the user. In normal registration, the host part of the Request-URI contains the IP address of the device and therefore, there is no way of knowing which registered user the INVITE is intended for. To overcome this issue, you can configure the device to use a special string with a unique value, "ac-feu=<value>" for each registration, allowing the device to differentiate between two registrations from the same user (identical REGISTER requests). Each REGISTER message is registered in the registration database using the unique "ac-feu=" string identifier for the Contact user.

A summary of how the device registers the two REGISTER messages in its registration database is as follows:

1. The addresses of the proxy servers that are configured on the SIP entity (IP Phone) must be the device's IP address with a different SIP local port for each one, for example:
   - Primary Proxy Server: 172.17.0.1:5060
   - Secondary Proxy Server: 172.17.0.1:5080
2. When the device receives two identical REGISTER messages from the SIP entity, it differentiates them by the SIP port on which they are received. The port allows the device to associate them with a SIP Interface (5060 for "Interface-1" and 5080 for "Interface-2").

3. The device performs SIP message manipulation (Pre-classification Manipulation) on the REGISTER messages to add a special parameter ("ac-int=<value>") to the Contact header to identify the SIP Interface on which each message is received. For example:
   - REGISTER for Primary Proxy received on SIP Interface "Interface-1":

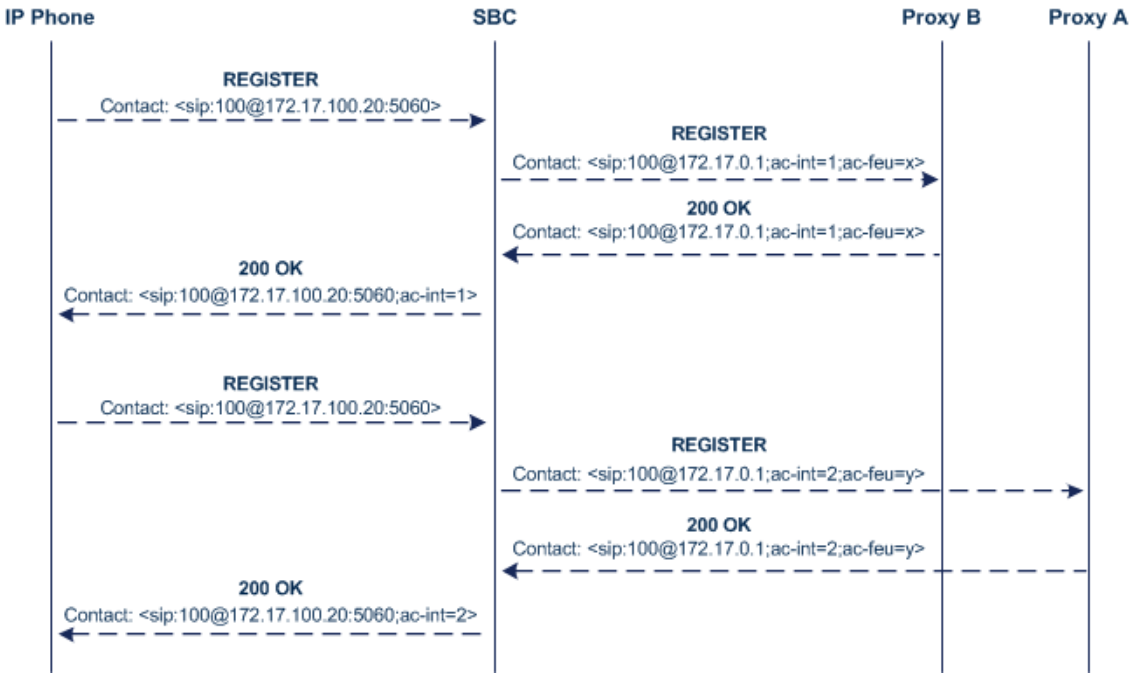
     ```
     REGISTER
     To: <sip:100@audc.com>
     Contact: <sip:100@172.17.100.20;ac-int=1>
     ```

   - REGISTER for Secondary Proxy received on SIP Interface "Interface-2":

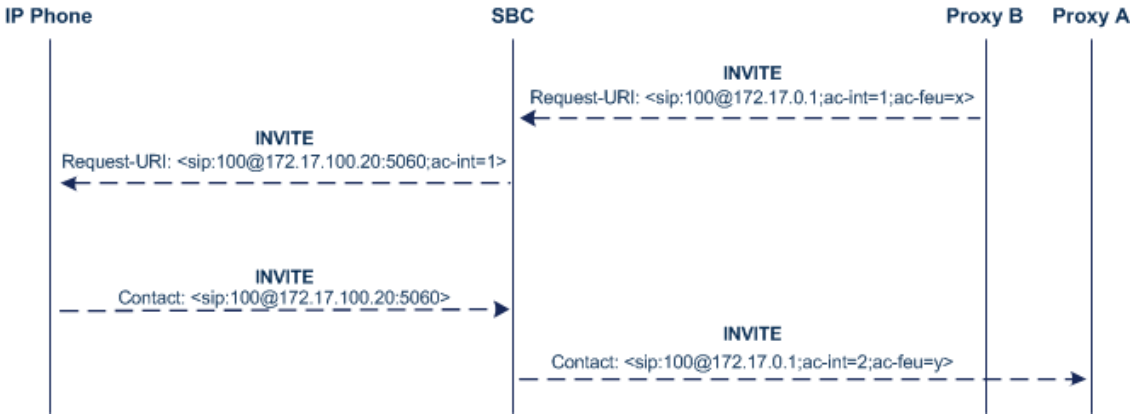
     ```
     REGISTER
     To: <sip:100@audc.com>
     Contact: <sip:100@172.17.100.20;ac-int=2>
     ```

4. The device adds to the Contact header a special string with a unique value, "ac-feu=<value>" for each registration (e.g., "Contact: <sip:100@172.17.100.20**;ac-int=1;ac-feu=x**>").
5. The device saves the two contacts (100@172.17.100.20;ac-int=1;ac-feu=x and 100@172.17.100.20;ac-int=2;ac-feu=y) under the **same AOR** (100@audc.com) in its user registration database.

The SIP call flow for dual registration is shown below:



The basic SIP call flow for INVITEs to and from the registered user is shown below:



➢ To configure support for dual registration:

1. On the SIP entity (IP Phone), configure the primary and secondary proxy server addresses as the IP address of the device and where each address has a different SIP port number.
2. Open the SBC General Settings page (**Setup** menu > **Signaling & Media** tab > **SBC** folder > **SBC General Settings**), and then from the 'Keep Original User in Register' drop-down list, select **Keep user; add unique identifier as URI parameter**.
3. In the Message Manipulations table, configure the following rules:
   - Index 0:
     - Manipulation Set ID: **1**
     - Action Subject: **header.contact.url.ac-int**
     - Action Type: **Modify**
     - Action Value: **'1'**
   - Index 1:
     - Manipulation Set ID: **2**
     - Action Subject: **header.contact.url.ac-int**
     - Action Type: **Modify**
     - Action Value: **'2'**
4. In the SIP Interfaces table, configure the following SIP Interfaces:
   - Index 0 (SIP Interface for IP Phone A):
     - Name: **Interface-1**
     - UDP Port: **5060**
     - Pre-classification Manipulation Set ID: **1**
   - Index 1 (SIP Interface for IP Phone B):
     - Name: **Interface-2**
     - UDP Port: **5080**

- Pre-classification Manipulation Set ID: **2**

5. In the Proxy Sets table, configure a Proxy Set for each proxy server (primary and secondary):
   - Index 0:
     - Proxy Name: **Primary**
     - SBC IPv4 SIP Interface: **Interface-1**
     - Proxy Address: **200.10.10.1**
   - Index 1:
     - Proxy Name: Secondary
     - SBC IPv4 SIP Interface: **Interface-2**
     - Proxy Address: **200.10.10.2**
6. In the IP Groups table, configure the following IP Groups:
   - Index 0:
     - Type: **Server**
     - Name: **Primary-Proxy**
     - Proxy Set: **Primary**
     - Classify By Proxy Set: **Enable**
   - Index 1:
     - Type: **Server**
     - Name: **Sec-Proxy**
     - Proxy Set: **Secondary**
     - Classify By Proxy Set: **Enable**
   - Index 2:
     - Type: **User**
     - Name: **IP-Phone-A**
   - Index 3:
     - Type: **User**
     - Name: **IP-Phone-B**
7. In the Classification table, configure rules to classify calls from the IP Phones based on SIP Interface:
   - Index 0:
     - Source SIP Interface: **Interface-1**

- Source IP Group: **IP-Phone-A**

- Index 1:
  - Source SIP Interface: **Interface-2**
  - Source IP Group: **IP-Phone-B**

8. In the IP-to-IP Routing table, configure the routing rules:
   - Index 0:
     - Source IP Group: **IP-Phone-A**
     - Destination IP Group: **Primary-Proxy**
   - Index 1:
     - Source IP Group: **Primary-Proxy**
     - Destination IP Group: **IP-Phone-A**
   - Index 2:
     - Source IP Group: **IP-Phone-B**
     - Destination IP Group: **Sec-Proxy**
   - Index 3:
     - Source IP Group: **Sec-Proxy**
     - Destination IP Group: **IP-Phone-B**

Handling Registered AORs with Same Contact URIs

The device can handle registration and call routing in cases where user registration includes AORs with the same Contact header URIs, as shown in the example below. Such a scenario typically occurs when two SIP endpoints reside in separate private networks and both are assigned the same local IP address.

- User 1 Registration:

```
REGISTER sip:300@10.33.4.140;user=phone SIP/2.0
Via: SIP/2.0/UDP 10.33.2.40;branch=OTGHREPCXDBIWECOCPJK
From: <sip:300@domain1;user=phone>;tag=ULYEYCGXHXMBPSOCXVWH
To: <sip:300@domain1;user=phone>
Call-ID: XDRXGAAWNVBTFHBMQCKE@10.33.2.38
```

```
CSeq: 1 REGISTER
Contact: <sip:300@10.33.2.40>
```

- User 2 Registration:

```
REGISTER sip:300@10.33.4.140;user=phone SIP/2.0
Via: SIP/2.0/UDP 10.33.2.40;branch=YHDWUJRMMOEIJRXVYKHD
From: <sip:300@domain2;user=phone>;tag=CVYTCHLIVMPBCGNGRTUA
To: <sip:300@domain2;user=phone>
Call-ID: INRNGFCHFHETRXAQNAIT@10.33.2.38
CSeq: 1 REGISTER
Contact: <sip:300@10.33.2.40>
```

For two such user registrations as shown in the example above, the device adds two AORs ("300@domain1" and "300@domain2") to its registration database, where each AOR is assigned the same Contact URI ("300@10.33.2.40"). To route a call to the correct user, the device needs to search the database for the full URI (user@host parts). To enable this support, perform the following configuration steps:

➢ **To enable handling of multiple AORs with identical Contact URIs:**

1. Open the Proxy & Registration page (**Setup** menu > **Signaling & Media** tab > **SIP Definitions** folder > **Proxy & Registration**).
2. From the 'DB Routing Search Mode' drop-down list (SBCDBRoutingSearchMode), select **Dest URI dependant**, and then click **Apply**.

Enabling Interworking of SIP and SIP-I Endpoints

The device can interwork between SIP and SIP-I endpoints for SBC calls. SIP-I endpoints are entities that are connected to the SS7 PSTN network, referred to as the ISDN User Part (ISUP) domain. The device supports the SIP-I Application-layer signaling protocol, which is the standard for encapsulating a complete copy of the SS7 ISUP message in SIP messages, according to ITU-T Q.1912.5, Interworking between Session Initiation Protocol (SIP) and Bearer

Independent Call Control protocol or ISDN User Part. In other words, SIP-I is SIP encapsulated with ISUP and the interworking is between SIP signaling and ISUP signaling. This allows you to deploy the device in a SIP environment where part of the call path involves the PSTN.

The SIP-I sends calls, originating from the SS7 network, to the SIP network by adding ISUP messaging in the SIP INVITE message body. The device can receive such a message from the SIP-I and remove the ISUP information before forwarding the call to the SIP endpoint. In the other direction, the device can receive a SIP INVITE message that has no ISUP information and before forwarding it to the SIP-I endpoint, create a SIP-I message by adding ISUP information in the SIP body. For SIP-I to SIP-I calls, the device can pass ISUP data transparently between the endpoints.

For the interworking process, the device maps between ISUP data (including cause codes) and SIP headers. For example, the E.164 number in the Request-URI of the outgoing SIP INVITE is mapped to the Called Party Number parameter of the IAM message, and the From header of the outgoing INVITE is mapped to the Calling Party Number parameter of the IAM message.

The ISUP data is included in SIP messages using the Multipurpose Internet Mail Extensions (MIME) body part, for example (some headers have been removed for simplicity):

```
INVITE sip:1774567@172.20.1.177;user=phone SIP/2.0
Via: SIP/2.0/UDP 172.20.73.230:5060;branch=z9hG4bK.iI
```

```
...
```

```
Accept: application/sdp, application/isup, applicatio
Content-Type: multipart/mixed; boundary=unique-bounda
MIME-Version: 1.0
Content-Length: 350
```

```
...
```

```
Content-Type: application/isup; version=FTSSURI; base
Content-Disposition: signal; handling=required
01 00 40 01 0a 02 02 08 06 83 10 71 47 65 07 08
01 00 00
--unique-boundary-1--
```

```
D6 SIP-T ISUP/IAM (Initial address message)
(--) len:-- >> Nature of connection indicators
Oct 1 : ---0---- Echo ctrl = Half echo not included
----00-- Cont. check = Not required
------00 Satellite = No circuit
(--) len:-- >> Forward call indicators
Oct 1 : 01------ ISUP pref. = Not req. all the way
--0----- ISUP indic. = Not used all the way
---0---- End-end inf = Not available
----0--- Interwork. = Not encountered
-----00- Method. ind = No method available
-------0 Call indic. = as National call
Oct 2 : -----00- SCCP method = No indication
```

ISUP data, received in the MIME body of the incoming SIP message is parsed according to the ISUP variant (SPIROU itu or ansi), indicated in the SIP Content-Type header. The device supports the following ISUP variants (configured by the 'ISUP Variant' parameter in the IP Profile table):

- French (France) specification, SPIROU (Système Pour l'Interconnexion des Réseaux OUverts), which regulates Telecommunication equipment that interconnect with networks in France. For SPIROU, the device sets the value of the SIP Content-Type header to "version=spirou; base=itu-t92+".
- ITU-92, where the device sets the value of the SIP Content-Type header to "version=itu-t92+; base=itu-t92+".

To configure interworking of SIP and SIP-I endpoints, using the 'ISUP Body Handling' parameter (IpProfile_SBCISUPBodyHandling) in the IP Profile table (see Configuring IP Profiles).

You can manipulate ISUP data, by configuring manipulation rules for the SIP Content-Type and Content-Disposition header values, in the Message Manipulations table (see Configuring SIP Message Manipulation). For a complete description of the ISUP manipulation syntax, refer to the *Syntax for SIP Message Manipulation Reference Guide*. In addition, you can use the AudioCodes proprietary SIP header X-AC-Action in Message Manipulation rules to support the various call actions (e.g., SIP-I SUS and RES messages) for the ISUP SPIROU variant. For more information, see Using the Proprietary SIP X-AC-Action Header.

Configuring SBC MoH from External Media Source

The External Media Source table lets you configure an external media (audio) source (streamer). The device can play Music-on-Hold (MoH) audio originating from this external media source, to SBC call parties that are placed on hold. Implementing an external media source offers flexibility in the type of audio that you want played as MoH (e.g., radio, adverts, or music). If you are not using an external media source, the device plays its' local default hold tone or a hold tone from an installed PRT file (depending on your configuration).

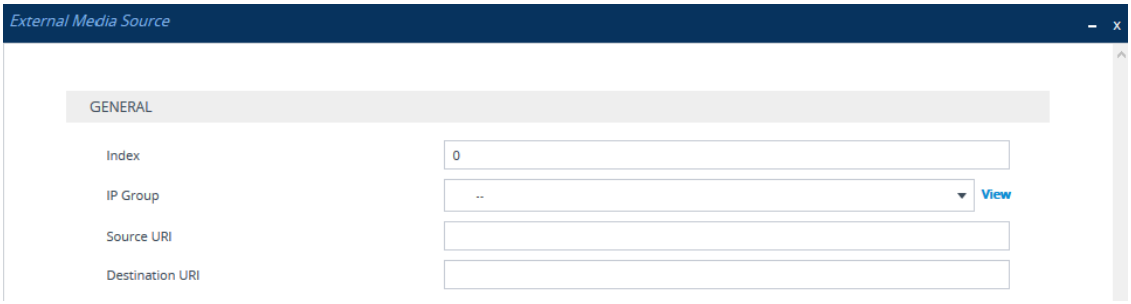
When a user (A) initiates call on-hold (i.e., sends a re-INVITE with SDP 'a=sendonly' or 'a=inactive' to the device), the device sends a new re-INVITE with SDP 'a=sendonly' to place the user (B) on hold. Once the user (B) responds with a SIP 200 OK, the device forwards the RTP audio stream for MoH from the external media source to the held party. When the user (A) retrieves the call (i.e., sends a re-INVITE with SDP 'a=sendrecv') to the held user (B), which then responds with a 200 OK, the device disconnects the held party from the external media source.

- Only one external media source can be connected to the device.
- The device can play MoH from an external media source to a maximum of 20 concurrent call sessions (on-hold parties).
- If you have configured an external media source and connection between the media source and the device is established, and you then modify configuration in this table, the device disconnects from the media source and then reconnects with it.
- If the connection with the media source is lost for any reason other than reconfiguration (e.g., receives a SIP BYE from the media source or RTP broken connection occurs), the device waits three seconds before attempting to re-establish the session by sending a new INVITE to the media source. This is repeated until the media source is reconnected or you disable the feature.

The following procedure describes how to configure an external media source through the Web interface. You can also configure it through ini file [ExternalMediaSource] or CLI (`configure voip > sbc external-media-source`).

➢ To configure an external media source:

1. Open the External Media Source table (**Setup** menu > **Signaling & Media** tab > **SBC** folder > **External Media Source**).
2. Click **New**; the following dialog box appears:



3. Configure the external media source according to the parameters described in the table below.
4. Click **Apply**; the device sends a SIP INVITE to the external media source and when SDP negotiation (e.g., for the offered coder) is complete and the device receives a SIP 200 OK response, connection is established and audio is continuously sent by the external media source to the device.

You can refresh the connection between the device and the external media source (mainly needed if you have modified configuration). When you do this, the device disconnects from the external media source and then reconnects with a new session.

➢ **To refresh connectivity:**

- On the table's toolbar, from the **Action** drop-down list, choose **Re-establish**.

Table 28-2: External Media Source Table Parameter Descriptions

Parameter	Description
'Index' [ExternalMediaSource_Index]	Defines an index number for the new table row.
'IP Group' `ip-group-name` [ExternalMediaSource_IPGroupName]	Assigns an IP Group from the IP Groups table (see Configuring IP Groups on page 451). This is the IP Group that represents the external audio streamer. **Note:** The parameter is mandatory.
'Source URI' `src-uri` [ExternalMediaSource_SourceURI]	Defines the source URI (user@host) of the SIP From header contained in the INVITE message that the device sends to the external media source. If you do not configure this parameter, the device sets the URI to the local IP address of the IP Interface on which the device sends the message.
'Destination URI' `dst-uri` [ExternalMediaSource_DestinationURI]	Defines the destination URI (user@host) of the SIP To header contained in the INVITE message that the device sends to the external media source. If you do not configure this parameter, the device sets the URI to the value of the IP Group's 'SIP Group Name' parameter.

Configuration of MoH from an external media source includes the following basic settings:

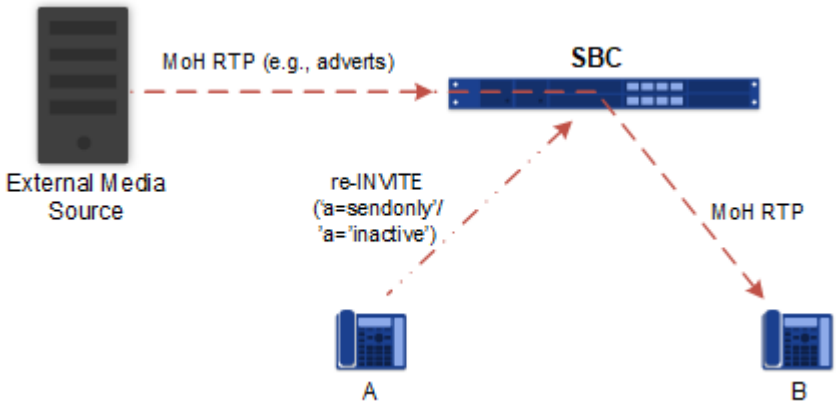
- Configuring an IP Profile (namely, the 'Extension Coders Group' parameter) and IP Group (namely, the 'IP Profile' parameter) for the media source
- Designating the media source IP Group as the external media source (in the External Media Source table, as described above)
- Configuring IP Profiles (namely, the 'Reliable Held Tone Source' and 'Play Held Tone' parameters) and IP Groups for the users

However, specific configuration may differ based on your implementation of this MoH feature. For example, you may implement this feature in one of the following architectures:

- A company with an on-site external media source for playing all MoH to branch users.

- A company with an on-site external media source that only plays MoH to branch users when connectivity with the remote media source is down

A configuration example of an on-site external media source that is always used to play MoH to its branch users is shown below and subsequently described.



1. Open the Coder Groups table (see Configuring Coder Groups on page 506), and then configure a Coders Group (e.g., AudioCodersGroups_0) with the coder(s) to use for communication between the device and the media source.
2. Open the IP Profiles table (see Configuring IP Profiles on page 519), and then configure two IP Profiles:
   - External Media Source:
     - 'Extension Coders Group': Assign the Coders Group configured in Step 1 (above).
   - Branch Users:
     - 'Reliable Held Tone Source': **No**
     - ' Play Held Tone': **External**
3. Open the IP Groups table (see Configuring IP Groups on page 451), and then configure two IP Groups:
   - External Media Source:
     - 'IP Profile': Assign the IP Profile configured for the external media source in Step 2 (above)
   - Branch Users:
     - 'IP Profile': Assign the IP Profile configured for the branch users in Step 2 (above)
4. Open the External Media Source table (see the beginning of this section), and then configure an External Media Source entity and associate it with the IP Group that you configured for the external media source in Step 3 (above).

WebRTC

The device supports interworking of Web Real-Time Communication (WebRTC) and SIP-based VoIP communication. The device interworks WebRTC calls made from a Web browser (WebRTC client) and the SIP destination. The device provides the media interface to WebRTC.

WebRTC is a browser-based real-time communication protocol. WebRTC is an open source, client-side API definition (based on JavaScript) drafted by the World Wide Web Consortium (W3C) that supports browser-to-browser applications for voice calling (video chat, and P2P file sharing) without plugins. Currently, the device's WebRTC feature is supported only by Mozilla Firefox and Google Chrome Web browsers other browsers are still not fully compatible with WebRTC). Though the WebRTC standard has obvious implications for changing the nature of peer-to-peer communication, it is also an ideal solution for customer-care solutions to allow direct access to the contact center. An example of a WebRTC application is a click-to-call button on a consumer Web site (see following figure). After clicking the button, the customer can start a voice and/or video call with a customer service personnel directly from the browser without having to download any additional software plugins. The figure below displays an example of a click-to-call application from a customer Web page, where the client needs to enter credentials (username and password) before placing the call.

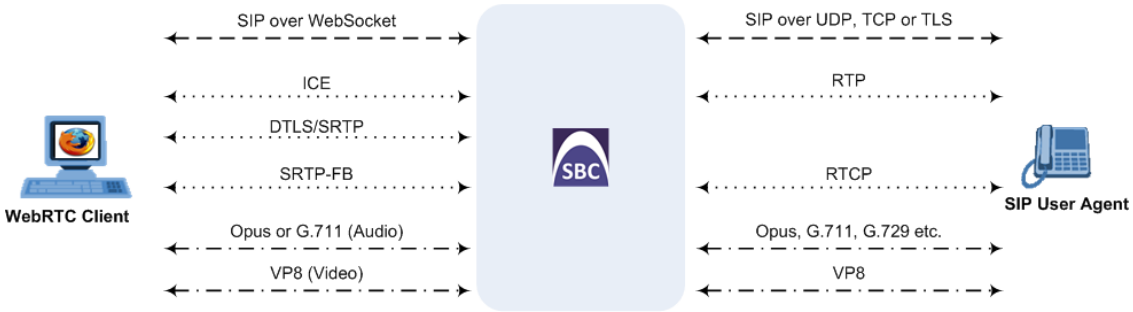
- The WebRTC feature is a license-based feature and is available only if it is included in the License Key that is installed on the device. For ordering the feature and the number of required WebRTC sessions, please contact the sales representative of your purchased device.
- For maximum concurrent WebRTC sessions (signaling-over-secure WebSocket and media-over-DTLS), refer to the device's *Release Notes*, which can be downloaded from AudioCodes website.

The WebRTC standard requires the following mandatory components, which are supported by the device:

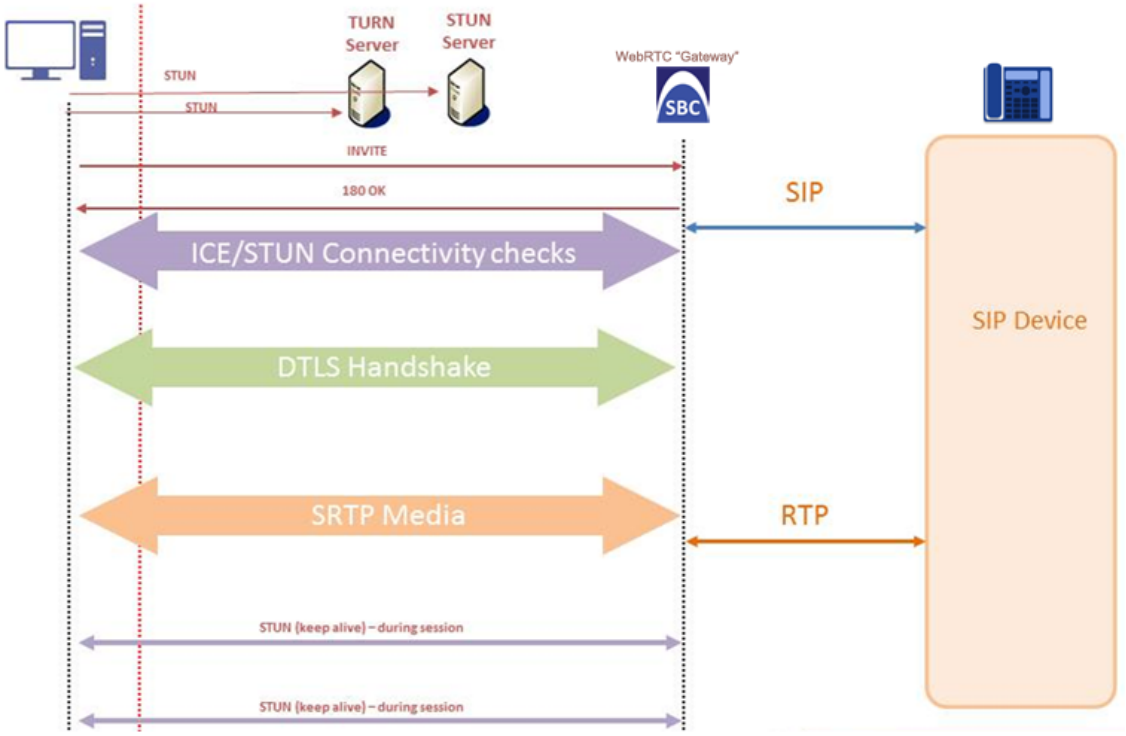
- **Voice coders:** Narrowband G.711 and wideband Opus (Version 1.0.3, per RFC 6176).

- **Video coders:** VP8 video coder. The device transparently forwards the video stream, encoded with the VP8 coder, between the endpoints.
- **ICE (per RFCs 5389/5245):** Resolves NAT traversal problems, using STUN and TURN protocols to connect peers. For more information, see ICE Lite.
- **DTLS-SRTP (RFCs 5763/5764):** Media channels must be encrypted (secured) through Datagram Transport Layer Security (DTLS) for SRTP key exchange. For more information, see SRTP using DTLS Protocol.
- **SRTP (RFC 3711):** Secures media channels by SRTP.
- **RTP Multiplexing (RFC 5761):** Multiplexing RTP data packets and RTCP control packets onto a single port for each RTP session. For more information, see Interworking RTP-RTCP Multiplexing.
- **Secure RTCP with Feedback (i.e., RTP/SAVPF format in the SDP - RFC 5124):** Combines secured voice (SRTP) with immediate feedback (RTCP) to improve session quality. The SRTP profile is called SAVPF and must be in the SDP offer/answer (e.g., "m=audio 11050 RTP/SAVP**F** 103"). For more information, see the IP Profile parameter, IPProfile_ SBCRTCPFeedback (see Configuring IP Profiles).
- **WebSocket:** WebSocket is a signaling (SIP messaging) transport protocol, providing full-duplex communication channels over a single TCP connection for Web browsers and clients. SIP messages are sent to the device over the WebSocket session. For more information, see SIP over WebSocket.

For more information on WebRTC, visit the WebRTC website at http://www.webrtc.org/. Below shows a summary of the WebRTC components and the device's interworking of these components between the WebRTC client and the SIP user agent:



The call flow process for interworking WebRTC with SIP endpoints by the device is illustrated below and subsequently described:



1. The WebRTC client uses a Web browser to visit the Web site page.
2. The Web page receives Web page elements and JavaScript code for WebRTC from the Web hosting server. The JavaScript code runs locally on the Web browser.
3. When the client clicks the Call button or call link, the browser runs the JavaScript code which sends the HTTP upgrade request for WebSocket in order to establish a WebSocket session with the device. The address of the device is typically included in the JavaScript code.
4. A WebSocket session is established between the WebRTC client and the device in order for the WebRTC client to register with the device. This is done using a SIP REGISTER message sent over the WebSocket session (SIP over WebSocket). Registration can be initiated when the client enters credentials (username and password) on the Web page or it can be done automatically when the client initially browses to the page. This depends on the design of the Web application (JavaScript). On the WebRTC client, the WebSocket connection is established for registration when the Web page is loaded; for click-to-call applications, registration is not needed and the WebSocket connection is established when the button for calling is clicked.
5. Once registered with the device, the client can receive or make calls, depending on the Web application.
6. To make a call, the client clicks the call button or link on the Web page.
7. Negotiation of a workable IP address between the WebRTC client and the device is done through ICE.
8. Negotiation of SRTP keys using DTLS is done between WebRTC and the client on the media.
9. Media flows between the WebRTC client and the SIP client located behind the device.

SIP over WebSocket

The device supports the transmission of SIP signaling over WebSocket. WebSocket is a protocol providing real-time, full-duplex (two-way) communication over a single TCP connection (socket) between a Web browser or page (client) and a remote host (server). This is used for browser-based applications such as click-to-call from a Web page. As WebSocket has been defined by the WebRTC standard as mandatory, its support by the device is important for deployments implementing WebRTC.

A WebSocket connection starts as an HTTP connection between the Web client and the server, guaranteeing full backward compatibility with the pre-WebSocket world. The protocol switch from HTTP to WebSocket is referred to as the WebSocket handshake, which is done over the same underlying TCP/IP connection. A WebSocket connection is established using a handshake between the Web browser (WebSocket client) and the server (i.e., the device). The browser sends a request to the server, indicating that it wants to switch protocols from HTTP to WebSocket. The client expresses its' desire through the Upgrade header (i.e., upgrade from HTTP to WebSocket protocol) in an HTTP GET request, for example:

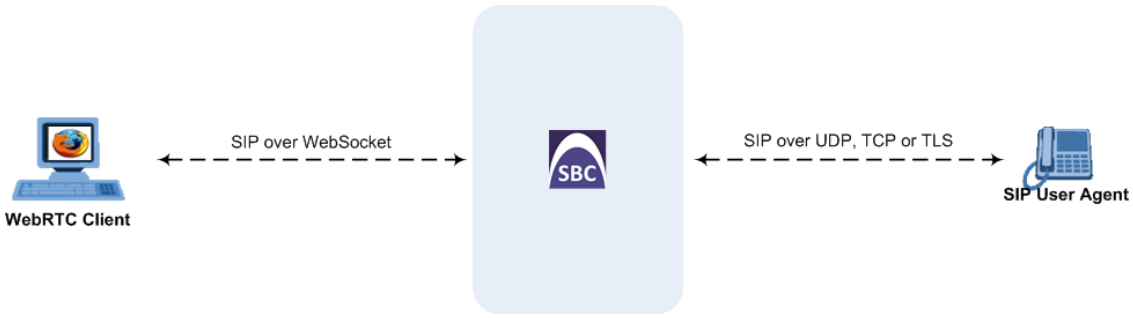
```
GET /chat HTTP/1.1
Upgrade: websocket
Connection: Upgrade
Host: <IP address:port of SBC device>
Sec-WebSocket-Protocol: SIP
Sec-WebSocket-Key: dGhlIHNhbXBsZSBub25jZQ==
Origin: <server that provided JavaScript code to browser, e.g., http://domain.com>
Sec-WebSocket-Version: 13
```

If the server understands the WebSocket protocol, it agrees to the protocol switch through the Upgrade header in an HTTP 101 response, for example:

```
HTTP/1.1 101 Switching Protocols
Upgrade: WebSocket
Connection: Upgrade
Sec-WebSocket-Accept: rLHCkw/SKsO9GAH/ZSFhBATDKrU=
Sec-WebSocket-Protocol: SIP
Server: SBC
```

At this stage, the HTTP connection breaks down and is replaced by a WebSocket connection over the same underlying TCP/IP connection. By default, the WebSocket connection uses the same ports as HTTP (80) and HTTPS (443).

Once a WebSocket connection is established, the SIP messages are sent over the WebSocket session. The device, as a "WebSocket gateway" or server can interwork WebSocket browser originated traffic to SIP over UDP, TCP or TLS, as illustrated below:



The SIP messages over WebSocket are indicated by the "ws" value, as shown in the example below of a SIP REGISTER request received from a client:

```
REGISTER sip:10.132.10.144 SIP/2.0
Via: SIP/2.0/WS v6iqlt8lne5c.invalid;branch=z9hG4bK7785666
Max-Forwards: 69
To: <sip:101@10.132.10.144>
From: "joe" <sip:101@10.132.10.144>;tag=ub50pqjgpr
Call-ID: fhddgc3kc3hhu32h01fghl
CSeq: 81 REGISTER
Contact: <sip:0bfr9fd5@v6iqlt8lne5c.invalid;transport=ws>;reg-
id=1;+sip.instance="<urn:uuid:4405bbe2-cf06-4c27-9c59-
6caf83af9b00>";expires=600
Allow: ACK,CANCEL,BYE,OPTIONS,INVITE,MESSAGE
Supported: path, outbound, gruu
User-Agent: JsSIP 0.3.7
Content-Length: 0
```

To keep a WebSocket session alive, it is sometimes necessary to send regular messages to indicate that the channel is still being used. Some servers, browsers or proxies may close an idle connection. Ping-Pong WebSocket messages are designed to send non-application level traffic that prevents the channel from being prematurely closed. You can configure how often the device pings the WebSocket client, using the [WebSocketProtocolKeepAlivePeriod] parameter (see Configuring WebRTC). The device always replies to ping control messages with a pong message. The WebSocket protocol supports keep-alive using special frames, however it is used only on the server side; for the Web client, a special ping (CRLF) request is used which the device answers.

In this way the client can detect connection failures

When the device operates in High-Availability (HA) mode, if a WebSocket connection has been established and a switchover subsequently occurs, the WebSocket session is not copied to the redundant device. As Chrome does not renew the WebSocket connection with the device, WebRTC calls remain open indefinitely; the Chrome side will stop the call, but the device will keep all of the call's resources open and the other side will have an active call with no voice. To prevent this, for the IP Profile associated with the WebRTC clients, configure the 'Broken Connection Mode' parameter to Disconnect.

Configuring WebRTC

To support WebRTC, you need to perform special configuration settings for the device's SBC leg interfacing with the WebRTC client (i.e., Web browser), as described in the following procedure.

For the WebRTC deployment environment, you need to install a signed certificate by a Certificate Authority (CA) on you Web server machine (hosting the WebRTC JavaScript) and on your AudioCodes SBC device (i.e., WebSocket server).

- Google announced a security policy change that impacts new versions of the Chrome Web browser. Any Web site that has integrated WebRTC, geolocation technology, screen-sharing and more, now requires to be served from a secure (HTTPS) site, including WebRTC-based WebSocket servers (WSS instead of WS). The configuration described below accommodates for this basic requirement.
- WebRTC JavaScript configuration is beyond the scope of this document.
- The device's WebRTC feature (*WebRTC Gateway*) can also operate with mobile device users that are registered to the device's WebRTC service, allowing them to make and receive WebRTC calls between registered users. For this support, you can use AudioCodes WebRTC client Software Development Kit (SDK) and Application Program Interface (API) to integrate the WebRTC functionality into the mobile applications (iOS and Android). For more information, refer to the:
  - ✓ *WebRTC iOS Client SDK API Reference Guide*
  - ✓ *WebRTC Android Client SDK API Reference Guide*
- For integrating the device's WebRTC functionality into client Web browsers for making calls from their Web browsers through the device, you can use AudioCodes WebRTC client Software Development Kit (SDK) and Application Program Interface (API). For more information, refer to the *WebRTC Web Browser Client SDK API Reference Guide*.
- You can implement the device's WebRTC widget, which can be embedded in websites and blogs without any previous knowledge of JavaScript. The widget creates a click-to-call button on your website. It can make calls to any user that is registered with the device. For more information, see the WebRTC Click-to-Call Widget Installation and Configuration Guide.

➢ To configure WebRTC:

1. Configure a TLS Context (certification):
   a. Open the TLS Contexts table (see Configuring TLS Certificate Contexts).

b. Add a new TLS Context (e.g., "WebRTC") or edit an existing one and configure the DTLS version (TLSContexts_DTLSVersion).

c. Create a certificate signing request (CSR) to request a digitally signed certificate from a Certification Authority (CA).

d. Send the CSR to the CA for signing.

e. When you have received the signed certificate, install it on the device as the "Device Certificate" and install the CA's root certificate into the device's trusted root store ("Trusted Certificates").

For more information on CSR, see Assigning CSR-based Certificates to TLS Contexts.

2. Configure the keep-alive interval with the WebSocket client:

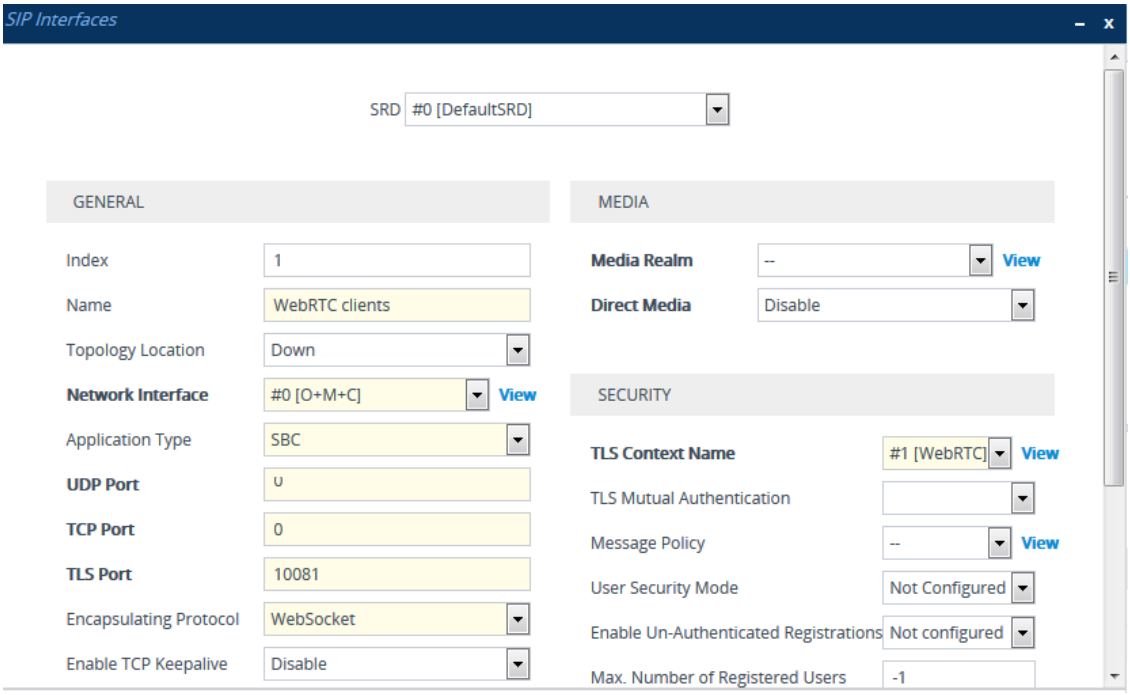
a. On the Transport Settings page (**Setup** menu > **Signaling & Media** tab > **SIP Definitions** folder > **Transport Settings**), and then in the 'WebSocket Keep-Alive Period' field (WebSocketProtocolKeepAlivePeriod), enter the keep-alive interval:

b. Click **Apply**.

3. Configure a SIP Interface for the WebRTC clients that identifies WebSocket traffic:

a. Open the SIP Interfaces table (see Configuring SIP Interfaces), and then configure the following:

- From the 'Encapsulating Protocol' drop-down list (SIPInterface_EncapsulatingProtocol), select **WebSocket**.
- In the 'TLS Port' field, configure the TLS port.
- From the 'TLS Context Name' drop-down list, assign the TLS Context that you configured in Step 1 (e.g., "WebRTC").



b. Click **Apply**.

4. Configure an IP Profile for the WebRTC clients:

a. Open the IP Profiles table (see Configuring IP Profiles), and then configure the following:

- From the 'SBC Media Security Mode' drop-down list (IpProfile_ SBCMediaSecurityBehaviour) , select **Secured**:

- From the 'SBC Media Security Method' drop-down list (IPProfile_ SBCMediaSecurityMethod), select **DTLS** to secure and encrypt media traffic through DTLS for SRTP key exchange:

- From the 'ICE Mode' drop-down list (IPProfile_SBCIceMode), select **Lite** to enable ICE:

- From the 'RTCP Mux' drop-down list (IPProfile_SBCRTCPMux), select **Supported** to enable RTCP multiplexing:

- From the 'RTCP Feedback' drop-down list (IPProfile_SBCRTCPFeedback), select **Feedback On** to enable RTCP feedback:

- From the 'Re-number MID' drop-down list (IpProfile_SBCRenumberMID), select **Enable** to enable the device to change the value of the 'a=mid:n' attribute (where *n* is a unique value) in the outgoing SDP offer (if the attribute is present) so that in the first media ('m=' line) the value will be 0, the next media the value will be 1, and so on.

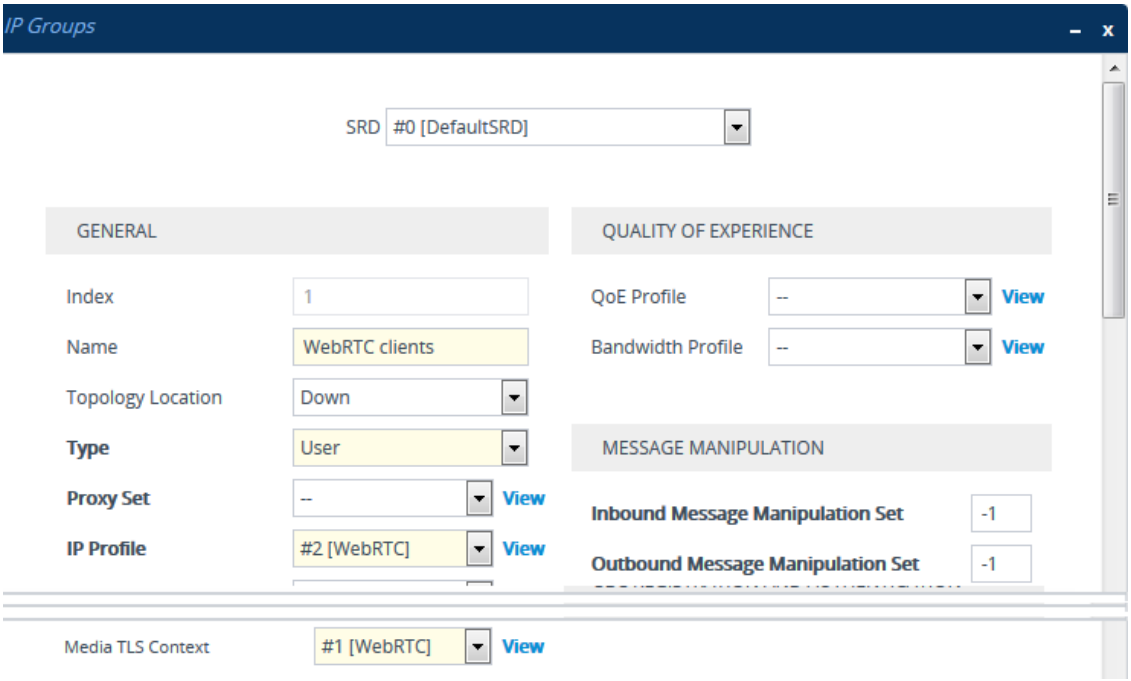
If the peer side also uses the 'mid' attribute in RTP extensions (e.g., a=extmap:3 urn:ietf:params:rtp-hdrext:sdes:mid), you also need to enable the 'Re-number MID' parameter for the IP Profile of the peer side.

- In the 'RFC 2833 DTMF Payload Type' parameter (IpProfile_SBC2833DTMFPayloadType), enter payload type "126":

- From the 'RTCP Mode' drop-down list (IPProfile_SBCRTCPMode), select **Transparent**:

b. Click **Apply**.

5. Configure an IP Group for the WebRTC clients:
   a. Open the IP Groups table (see Configuring IP Groups).
   b. Do the following:
   - From the 'Type' drop-down list, select **User**.
   - From the 'IP Profile' drop-down list, select the IP Profile that you configured for the WebRTC clients in Step 3 (e.g., "WebRTC").
   - From the 'Media TLS Context' drop-down list, select the TLS Context that you configured in Step 1. For more information on DTLS, see SRTP using DTLS Protocol.



6. Configure IP-to-IP routing rules to route calls between the WebRTC clients and the enterprise:
   a. Open the IP-to-IP Routing table (see Configuring SBC IP-to-IP Routing Rules).
   b. Configure routing rules for the following call scenarios:
      - Call routing from WebRTC clients (IP Group configured in Step 4) to the enterprise.
      - Call routing from the enterprise to the WebRTC clients (IP Group configured in Step 4).
7. Enable the device to include all previously negotiated media lines ('m=') in the SDP offer-answer exchanges for the WebRTC session:
   a. Open the Media Settings page (**Setup** menu > **Signaling & Media** tab > **Media** folder > **Media Settings**).
   b. Under the SBC Settings group, from the 'Enforce Media Order' drop-down list (SBCEnforceMediaOrder), select **Enable**:

   c. Click **Apply**.

Call Forking

This section describes various Call Forking features supported by the device.

Initiating SIP Call Forking

The SBC device supports call forking of an incoming call to multiple SBC users (destinations). Call forking is supported by the device's capability of registering multiple SIP client user phone

contacts (mobile and fixed-line extensions) under the same Address of Record (AOR) in its registration database. This feature can be implemented in the following example scenarios:

- An enterprise Help Desk, where incoming customer calls are simultaneously sent to multiple customer service agent extensions.
- An employee's phone devices, where the incoming call is simultaneously sent to multiple devices (e.g., to the employee's office phone and mobile SIP phone).
- An enterprise reception desk, where an incoming call is simultaneously sent to multiple receptionists.

The device supports various modes of call forking. For example, in Parallel call forking mode, the device sends the INVITE message simultaneously to all the users registered under the same AOR, resulting in the ringing of all extensions; the first extension to pick up the call receives the call, and all other extensions stop ringing. The Call Forking feature is configured by creating a User-type IP Group and configuring the IP Groups table's parameter, 'SBC Client Forking Mode' (see Configuring IP Groups).

The device can also fork INVITE messages received for a Request-URI of a specific contact (user), belonging to the destination IP Group User-type, registered in the database to all other users located under the same AOR as the specific contact. This is configured by the [SBCSendInviteToAllContacts] parameter.

Configuring SIP Forking Initiated by SIP Proxy

The device can handle the receipt of multiple SIP 18x responses as a result of SIP forking initiated by a proxy server. This occurs when the device forwards an INVITE, received from a user agent (UA), to a proxy server and the proxy server then forks the INVITE request to multiple UAs. Several UAs may answer and the device may therefore, receive several replies (responses) for the single INVITE request. Each response has a different 'tag' value in the SIP To header.

During call setup, forked SIP responses may result in a single SDP offer with two or more SDP answers. The device "hides" all the forked responses from the INVITE-initiating UA, except the first received response ("active" UA) and it forwards only subsequent requests and responses from this active UA to the INVITE-initiating UA. All requests/responses from the other UAs are handled by the device; SDP offers from these UAs are answered with an "inactive" media.

The device supports two forking modes:

- **Latch On First:** The device forwards only the first received 18x response to the INVITE-initiating UA and disregards subsequently received 18x forking responses (with or without SDP).
- **Sequential:** The device forwards all 18x responses to the INVITE-initiating UA, sequentially (one after another). If 18x arrives with an offer only, only the first offer is forwarded to the INVITE-initiating UA.

➢ **To configure the call forking mode:**

1. Open the SBC General Settings page (**Setup** menu > **Signaling & Media** tab > **SBC** folder > **SBC General Settings**).
2. From the 'Forking Handling Mode' [SBCForkingHandlingMode] drop-down list, select the required mode:

3. Click **Apply**.

The device also supports media synchronization for call forking. If the active UA is the first one to send the final response (e.g., 200 OK), the call is established and all other final responses are acknowledged and a BYE is sent if needed. If another UA sends the first final response, it is possible that the SDP answer that was forwarded to the INVITE-initiating UA is irrelevant and thus, media synchronization is needed between the two UAs. Media synchronization is done by sending a re-INVITE request immediately after the call is established. The re-INVITE is sent without an SDP offer to the INVITE-initiating UA. This causes the INVITE-initiating UA to send an offer which the device forwards to the UA that confirmed the call. Media synchronization is enabled by the EnableSBCMediaSync parameter.

Configuring Call Forking-based IP-to-IP Routing Rules

You can configure call forking routing rules in the IP-to-IP Routing table. This is done by configuring multiple routing rules under a forking group. These rules send an incoming IP call to multiple destinations of any type (e.g., IP Group or IP address). The device forks the call by sending simultaneous INVITE messages to all the specified destinations. It handles the multiple SIP dialogs until one of the calls is answered and then terminates the other SIP dialogs. For more information, see Configuring SBC IP-to-IP Routing Rules.

Call Survivability

This section describes various call survivability features supported by the SBC device.

Enabling Auto-Provisioning of Subscriber-Specific Information of BroadWorks Server for Survivability

This feature enables SBC user registration for interoperability with BroadSoft BroadWorks server to provide call survivability in case of connectivity failure with the BroadWorks server, for example, due to a WAN failure. The feature enables local users to dial a local extension (or any other configured alias) that identifies another local user, in survivability mode.

In normal operation, when subscribers (such as IP phones) register with the BroadWorks server through the device, the device includes the SIP Allow-Events header in the sent REGISTER message. In response, the BroadWorks server sends the device a SIP 200 OK containing an XML body with subscriber information such as extension number, phone number, and URIs (aliases), as shown in the example below:

```
<?xml version="1.0" encoding="utf-8"?>
 <BroadsoftDocument version="1.0" content="subscriberData">
  <phoneNumbers>
   <phoneNumber>2403645317</phoneNumber>
   <phoneNumber>4482541321</phoneNumber>
  </phoneNumbers>
  <aliases>
   <alias>sip:bob@broadsoft.com</alias>
   <alias>sip:rhughes@broadsoft.com</alias>
  </aliases>
  <extensions>
   <extension>5317</extension>
   <extension>1321</extension>
  </extensions>
 </BroadSoftDocument>
```

The device forwards the 200 OK to the subscriber (without the XML body). The call flow is shown below:

The device saves the users in its registration database with their phone numbers and extensions, enabling future routing to these destinations during survivability mode when communication with the BroadWorks server is lost. When in survivability mode, the device routes the call to the Contact associated with the dialed phone number or extension number in the registration database.

➢ **To enable the BroadWorks survivability feature:**

1. Open the SBC General Settings page (**Setup** menu > **Signaling & Media** tab > **SBC** folder > **SBC General Settings**).
2. From the 'BroadWorks Survivability Feature' drop-down list (SBCExtensionsProvisioningMode), select **Enable**:

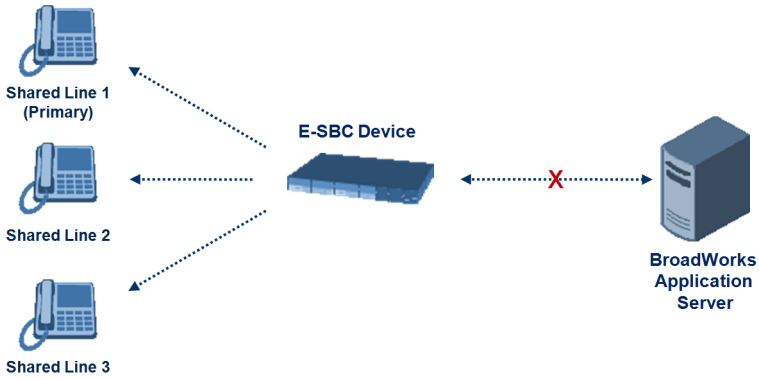
3. Click **Apply**.

Configuring BroadSoft's Shared Phone Line Call Appearance for Survivability

The device can provide redundancy for BroadSoft's Shared Call Appearance feature. When the BroadSoft application server switch (AS) fails or does not respond, or when the network

connection between the device and the BroadSoft AS is down, the device manages the Shared Call Appearance feature for the SIP clients.

The feature is supported by configuring a primary extension and associating it with secondary extensions (i.e., *shared lines*) so that incoming calls to the primary extension also ring at the secondary extensions. The call is established with the first extension to answer the call and consequently, the ringing at the other extensions stop. For example, assume primary extension number 600 is shared with secondary extensions 601 and 602. In the case of an incoming call to 600, all three phone extensions ring simultaneously, using the device's call forking feature as described in Configuring SIP Forking Initiated by SIP Proxy. Note that incoming calls specific to extensions 601 or 602 ring only at these specific extensions.



To configure this capability, you need to configure a shared-line, inbound manipulation rule for registration requests to change the destination number of the secondary extension numbers (e.g. 601 and 602) to the primary extension (e.g., 600). Call forking must also be enabled. The following procedure describes the main configuration required.

- The device enables outgoing calls from all equipment that share the same line simultaneously (usually only one simultaneous call is allowed per a specific shared line).
- You can configure whether REGISTER messages from secondary lines are terminated on the device or forwarded transparently (as is), using the SBCSharedLineRegMode parameter.
- The LED indicator of a shared line may display the wrong current state.

➢ To configure BroadSoft's Shared Line feature:

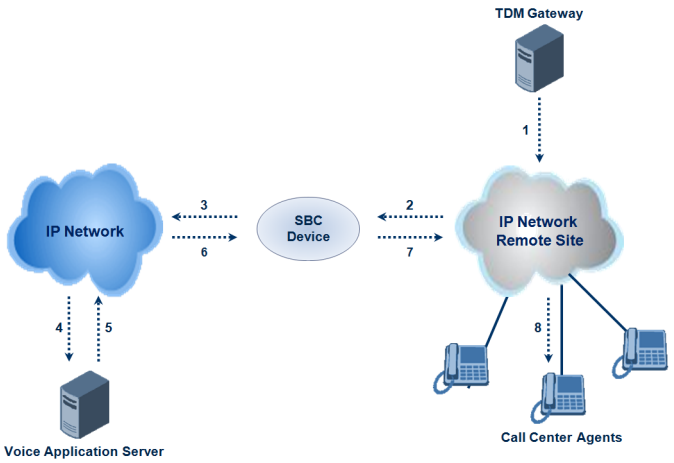
1. In the IP Groups table (see Configuring IP Groups), add a Server-type IP Group for the BroadWorks server.
2. In the IP Groups table, add a User-type IP Group for the IP phone users and set the 'SBC Client Forking Mode' parameter to **Parallel** so that the device forks incoming calls to all contacts under the same AOR registered in the device's registration database.
3. In the IP-to-IP Routing table (see Configuring SBC IP-to-IP Routing Rules), add a rule for routing calls between the above configured IP Groups.

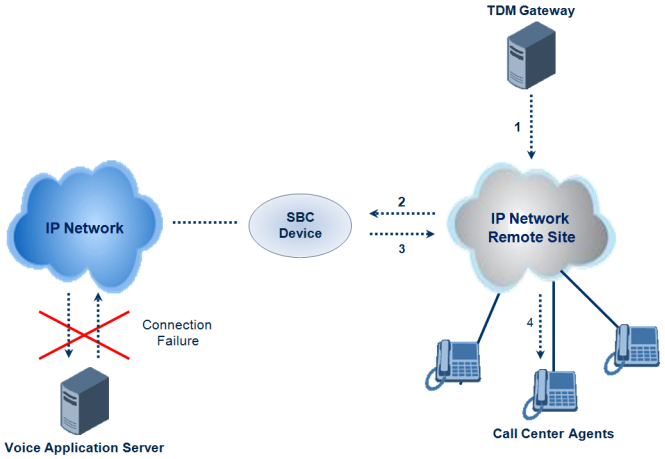
4. In the Inbound Manipulations table (see Configuring IP-to-IP Inbound Manipulations), add a manipulation rule for the secondary extensions (e.g., 601 and 602) so that they also register to the device's database under the primary extension contact (e.g., 600):
   - 'Manipulation Purpose': **Shared Line**
   - Match:
     - 'Request Type': **REGISTER**
     - 'Source IP Group': IP Group that you created for the users (e.g., 2)
     - 'Source Username Pattern': Represents the secondary extensions, e.g., 601 and 602
   - Action:
     - 'Manipulated URI': **Source** (manipulates the source URI)
     - 'Remove From Right': 1 (removes the last digit of the extensions, e.g., 601 is changed to 60)
     - 'Suffix to Add': 0 (adds 0 to the end of the manipulated number, e.g., 60 is changed to 600)

Configuring Call Survivability for Call Centers

The device supports call survivability for call centers. When a communication failure (e.g., in the network) occurs with the remote voice application server responsible for handling the call center application (such as IVR), the device routes the incoming calls received from the customer (i.e., from the TDM gateway) to the call center agents.

In normal operation, the device registers the agents in its users registration database. Calls received from the TDM gateway are forwarded by the device to the application server, which processes the calls and sends them to specific call center agents, through the device. Upon a failure with the application server, the device routes the calls from the TDM Gateway to the agents. The device routes the call to the first available user it founds. If the call is not answered by the user, the device routes it to the next available user. The SBC can handle a sequence of up to five users, after which the session is timed out and the call is dropped.

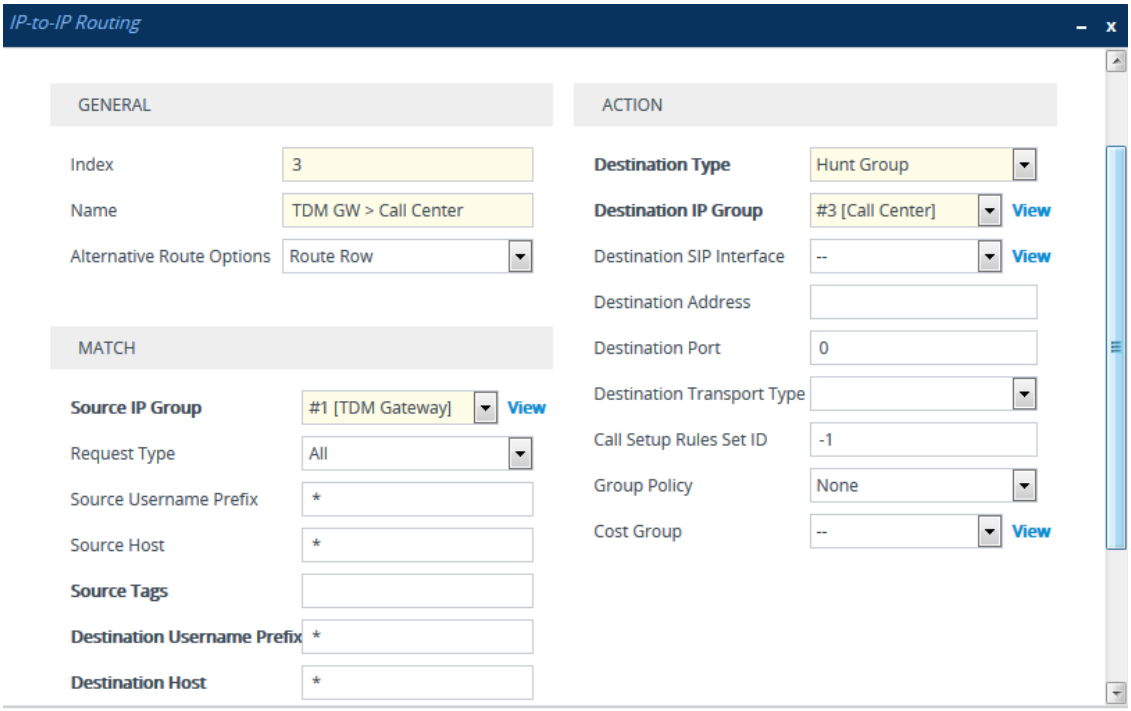




➢ To configure call survivability for a call center application:

1. In the IP Groups table (see Configuring IP Groups), add IP Groups for the following entities:
   - TDM Gateway (Server-type IP Group). This entity forwards the customer calls through the device to the Application server.
   - Application server (Server-type IP Group). This entity processes the call and sends the call through the device to the specific call center agent located on a different network (remote).
   - Call center agents (User-type IP Group). You can configure multiple IP Groups to represent different groups of call center agents, for example, agents and managers.
2. In the Classification table (see Configuring Classification Rules), add rules to classify incoming calls that are received from the entities listed in Step 1, to IP Groups.
3. In the SBC IP-to-IP Routing table (see Configuring SBC IP-to-IP Routing Rules), add the following IP-to-IP routing rules:
   - For normal operation:
     - Routing from TDM Gateway to Application server.
     - Routing from Application server to call center agents.
   - For call survivability mode: Routing from TDM Gateway to call center agents. This configuration is unique due to the following settings:
     - The 'Source IP Group' field is set to the IP Group of the TDM Gateway.
     - The 'Destination Type' field is set to **Hunt Group**, which is specifically used for call center survivability.
     - The 'Destination IP Group' field is set to the IP Group of the call center agents.

   The figure below displays a routing rule example, assuming IP Group #1 represents the TDM Gateway and IP Group #3 represents the call center agents:



Enabling Survivability Display on Aastra IP Phones

If the SBC device is deployed in a network with Aastra IP phones and connectivity with the WAN fails, the device provides call survivability by enabling communication between IP phone users within the LAN enterprise. In such a scenario, the device can be configured to notify the IP phones that it is currently operating in Survivability mode. When this occurs, the Aastra IP phones display the message, "Stand Alone Mode" on their LCD screens.

If you enable the feature and the device is in Survivability mode, it responds to SIP REGISTER messages from the IP phones with a SIP 200 OK containing the following XML body:

```
Content-Type: application/xml
<?xml version="1.0" encoding="utf-8"?>
<LMIDocument version="1.0">
<LocalModeStatus>
  <LocalModeActive>true</LocalModeActive>
  <LocalModeDisplay>StandAlone Mode</LocalModeDisplay>
</LocalModeStatus>
</LMIDocument>
```

➢ **To enable survivability display on Aastra phones:**

1. Load an ini file to the device that includes the following parameter setting:

```
SBCEnableSurvivabilityNotice = 1
```

Alternative Routing on Detection of Failed SIP Response

The device can detect failure of a sent SIP response (e.g., TCP timeout, and UDP ICMP). In such a scenario, the device re-sends the response to an alternative destination. This support is in addition to alternative routing if the device detects failed SIP requests.

For example, assume the device sends a SIP 200 OK in response to a received INVITE request. If the device does not receive a SIP ACK in response to this, it sends a new 200 OK to the next alternative destination. This new destination can be the next given IP address resolved from a DNS from the Contact or Record-Route header in the request related to the response.

Configuring Push Notification Service

The device supports the Push Notification Service per IETF draft "Push Notification with the Session Initiation Protocol (SIP)". This service is used to wake end-user equipment (typically, mobile platforms) and operating systems that have gone to "sleep" (to save resources such as battery life) so that they can receive traffic. Typically, each operating system uses a dedicated Push Notification Service. For example, Apple iOS devices use the Apple Push Notification service (APNs) while Android devices use the Firebase Cloud Messaging (FCM) service. Without using a Push Notification Service to wake SIP User Agents (UAs), UAs wouldn't be able to send binding-refresh SIP REGISTER requests, receive SIP requests (e.g., INVITE), or send periodic keep-alive messages for maintaining connectivity with SIP servers. The device communicates with third-party, HTTP-based Push Notification Servers over HTTP, using RESTful APIs for exchanging information (currently, only JSON format is supported).

SIP users wanting to receive push notifications must specify the following parameters in the Contact header of the SIP REGISTER request that it sends to the device for registration:

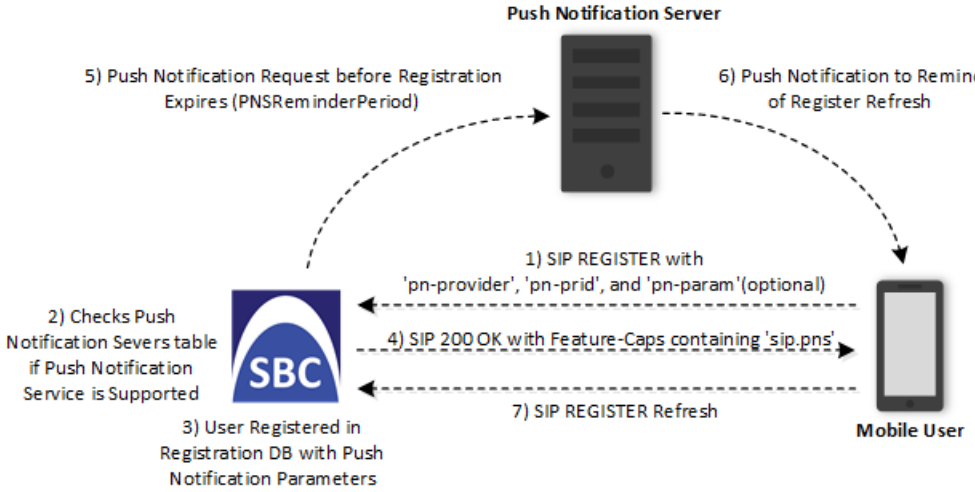
- 'pn-provider': Specifies the type of Push Notification Service.
- 'pn-prid': Specifies the unique identifier (Push Resource ID / PRID) that the Push Notification Service uses to identify the user.
- 'pn-param': (Optional) Specifies additional implementation-specific data required by the Push Notification Service.

Below shows an example of a REGISTER message containing the Push Notification parameters (in bold):

```
REGISTER sip:alice@example.com SIP/2.0
Via: SIP/2.0/TCP alicemobile.example.com:5060;branch=z9hG4bKnashds7
Max-Forwards: 70
To: Alice <sip:alice@example.com>
From: Alice <sip:alice@example.com>;tag=456248
Call-ID: 843817637684230@998sdasdh09
CSeq: 1826 REGISTER
Contact: <sip:alice@alicemobile.example.com;
pn-provider=acme;
```

```
pn-param=acme-param;
pn-prid=ZTY4ZDJlMzODE1NmUgKi0K>
Expires: 7200
Content-Length: 0
```

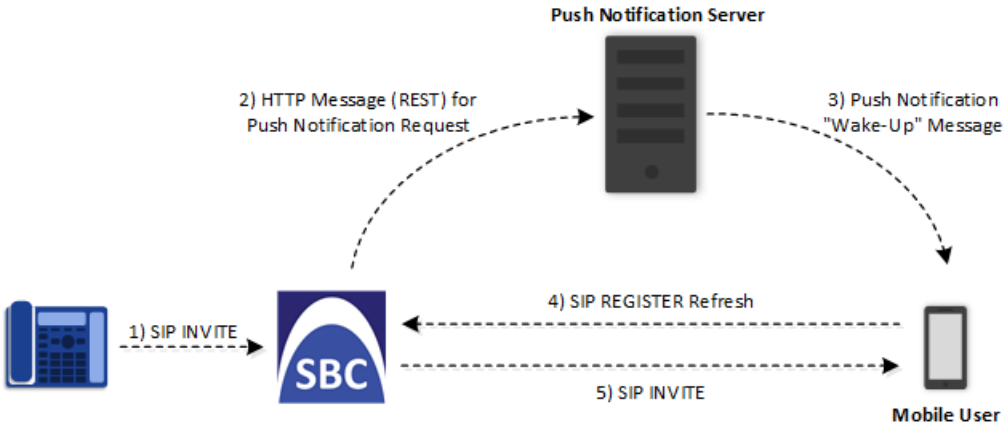
The device handles registrations from users requiring Push Notification Service, as follows:



1. The user sends a REGISTER request to the device that contains the Push Notification parameters in the Contact header, as mentioned previously.
2. The device searches its Push Notification Servers table for a row whose 'Provider' parameter has the * wildcard character value, or the same value as the 'pn-provider' parameter in the REGISTER request.
3. Regardless of the search, the device adds the user in its registration database with the Push Notification parameters (mentioned previously).
4. If a matching row in the Push Notification Servers table is located, the device sends a SIP 200 OK response containing the Feature-Caps header with the 'sip.pns=' feature-capability indicator, identifying the type of Push Notification Service as specified in the 'pn-provider' parameter (e.g., Feature-Caps: *;+**sip.pns="acme"**;+sip.pnsreg="121"). If no matching row in the table is located (i.e., Push Notification Service is not supported), the device sends a 200 OK response, but without the Feature-Caps header.
5. (6 and 7) At a user-defined time (using the [PNSReminderPeriod] parameter) before the user's registration expires, the device sends a push notification request containing the user's PRID to the Push Notification Server to trigger it into sending a push notification to the user to remind it to send a refresh REGISTER message to the device.

If the user sends the device a refresh REGISTER request without the Push Notification parameters, the device considers the user as no longer using Push Notification Service. In this scenario, the device stops sending push notification requests to the Push Notification Server for the user.

Once a user is registered with the device, the device can route calls to it. The following figure shows how the device processes an incoming dialog-initiating SIP request (e.g., INVITE) whose destination is a mobile user that uses Push Notification Service:



6. The device receives an incoming call (SIP INVITE message) for the mobile user, which according to the device's registration database (i.e., user's registration includes Push Notification parameters), uses a Push Notification Service.
7. The device sends a push notification request containing the user's PRID (over HTTP) to the Push Notification Server. The device uses the Push Notification Servers table to determine which Push Notification Server to send this push notification request. The device searches the table for a row that is configured with the value of the user's 'pn-provider' parameter (table's 'Provider' parameter) and if located, sends the push notification request to the address of the associated Remote Web Service.
8. The Push Notification Server sends a push notification to the user to "wake" it up.
9. The user sends a refresh SIP REGISTER message to the device, which indicates that the user is "awake" and ready to receive the call.
10. The device sends the INVITE message to the user, using its regular routing logic.

- If the push notification request that is sent to the Push Notification Server fails, the device rejects the INVITE message with a SIP 480 response.
- If the device doesn't receive a refresh REGISTER message within a user-defined time (configured by the [PNSRegisterTimeout] parameter), the device rejects the INVITE with a SIP 480 response.
- When the device receives an incoming INVITE message for a user who is registered for push notification, but the corresponding row in the Push Notification Servers table has been deleted, the device immediately forwards the INVITE message to the user (as though the user had not requested push notification service).

➢ **To configure Push Notification Service:**

1. Configure a Remote Web Service (see Configuring Remote Web Services on page 316) to represent the HTTP-based Push Notification Server (address and other required

parameters). You must configure the Remote Web Service with the 'Type' parameter set to **General**.

2. Configure the Push Notification service in the Push Notification Servers table (see Configuring Push Notification Servers on page 631). This table configures the Push Notification Service type, the Remote Web Service that you configured in Step 1, and the information-exchange protocol (currently, only JSON) used between the device and the server. Therefore, the device uses this table to determine which Push Notification Server to send push notification requests for a specific user. The device searches the table for a row that is configured with the value of the user's 'pn-provider' parameter (table's 'Provider' parameter) and if located, sends the push notification request to the Push Notification Sever using the address of the associated Remote Web Service.
3. Configure the time (in seconds) before the user's registration on the device expires, when the device sends a push notification request (over HTTP) to the Push Notification Server to trigger it into sending a push notification to the user to remind it to send a refresh REGISTER message to the device. This is configured by the [PNSReminderPeriod] parameter or CLI command `configure voip > sbc settings > pns-reminder-period`).
4. Configure the time (in seconds) that the device must wait for a refresh REGISTER message from the user after the device sends a push notification request to the Push Notification Server for the user, when the device receives an incoming SIP dialog-initiating request (e.g., INVITE) that it must send to the user. This is configured by the [PNSRegisterTimeout] parameter or CLI command `configure voip > sbc settings > pns-register-timeout`.

VoIPerfect

The device's VoIPerfect™ feature combines Access and Enterprise SBC technology to ensure high speech (call) quality (MOS) between the Enterprise SBC and the Access SBC (located at the Internet service provider / ISP) during periods of adverse WAN network conditions (such as packet loss and bandwidth reduction). VoIPerfect adapts itself to current network conditions. Before adverse WAN network conditions can affect the quality of the call, VoIPerfect employs sophisticated technology using the Opus coder or G.729 (explained later in this section) to ensure that high call quality is maintained.

VoIPerfect guarantees that 95% of your calls will achieve a Perceptual Evaluation of Speech Quality (PESQ) score greater than or equal to 3.6 if the summation of bandwidth overuse and packet loss is less than or equal to 25%. . However, for VoIPerfect with G.729 (Managed G.729, discussed later) operating in an MPLS environment, this PESQ score is achieved if bandwidth overuse is less than or equal to 50%. ISPs can therefore offer service level agreements (SLAs) to their customers based on the VoIPerfect feature. For more information, contact the sales representative of your purchased device. In addition, by ensuring high call quality even in adverse network conditions, VoIPerfect may reduce costs for ISPs such as SIP trunk providers and Unified Communications as a Service (UCaaS) by eliminating the need for dedicated WAN

links (such as MPLS and leased links) and instead, allow the use of standard broadband Internet connections. However, it can also be used in tandem with existing infrastructure.

VoIPerfect uses Temporary Maximal Media Stream Bit Rate (TMMBR) negotiation capabilities for Opus coders. Through TMMBR, VoIPerfect can receive indications of network quality and dynamically change the coder's payload bit rate accordingly during the call to improve voice quality. TMMBR is an RTCP feedback message (per RFC 4585) which enables SIP users to exchange information regarding the current bit rate of the media stream. The information can be used by the receiving side to change the media stream parameters (e.g., coder rate or coder) to enhance voice quality. TMMBR is negotiated in the SDP Offer/Answer model using the 'tmbr' attribute and following syntax:

```
a=rtcp-fb:<payload type> ccm tmmbr smaxpr=<sent TMMBR packets)
```

VoIPerfect also supports the SDP attribute 'a=rtcp-rsize', which reduces the RTCP message size (RFC 5506). As feedback messages are frequent and take a lot of bandwidth, the attribute attempts to reduce the RTCP size. The attribute can only be used in media sessions defined with the AVPF profile and must also be included in sessions supporting TMMBR; otherwise, the call is rejected.

VoIPerfect supports two modes of operation, where the Access SBC can be configured to support both modes and each Enterprise SBC serviced by the Access SBC can be configured to support one of the modes:

- **Managed Opus or Managed G.729:** If the SBC detects WAN network impairments during a call using the Opus or G.729 coder between the Enterprise SBC and Access SBC, it can adjust the coder's attributes (e.g., bit rate) for that specific call to ensure high voice quality is maintained. The advantage of these coders is that their bit rate can change dynamically according to bandwidth availability. This mode is useful for unstable networks, allowing the coders to dynamically adapt to adverse network conditions.

  For Managed Opus, the Enterprise SBC performs transcoding from G.711 (used between Enterprise phones and SBC) to Opus (used between Enterprise and Access SBCs). For Managed G.729, the G.729 coder is used by all the involved entities and therefore, transcoding is not needed. For Managed G.729 operating in an MPLS environment, voice quality can also be maintained, as mentioned previously.

Figure 28-1: VoIPerfect Managed Opus

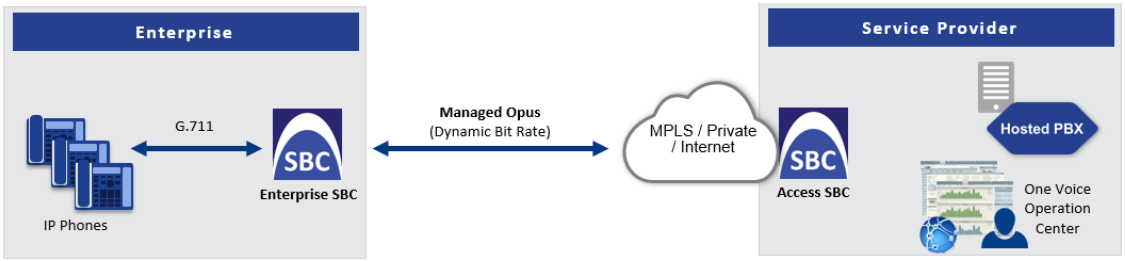
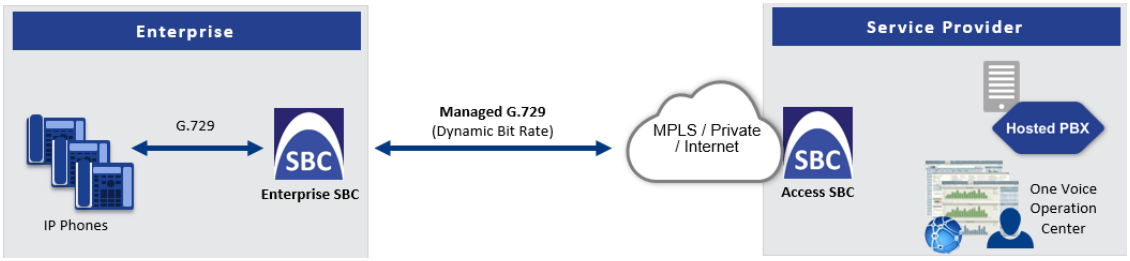


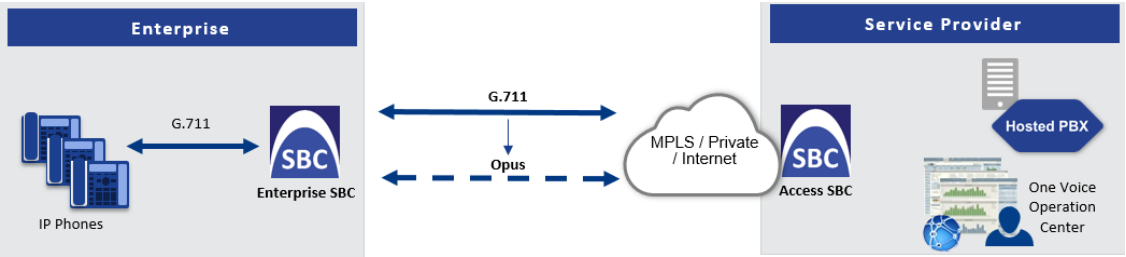
Figure 28-2: VoIPerfect Managed G.729



Configuration of the Enterprise SBC:

- Coder Groups table (see Configuring Coder Groups): Coder Group with Opus or G.729 (depending on Managed coder)
- Allowed Audio Coders Groups table (see Configuring Allowed Audio Coder Groups): Allowed Audio Coder Group with Opus or G.729 (depending on Managed coder)
- IP Profiles table (see Configuring IP Profiles):
  - (Managed Opus Only) 'Extension Coders Group': Select the Coders Group for Opus
  - 'Allowed Audio Coders': Select the Allowed Audio Coders Group with Opus or G.729 (depending on Managed coder)
  - 'Allowed Coders Mode': **Restriction**
  - 'Voice Quality Enhancement': **Enable**
  - 'RTCP Feedback': **Feedback On**
  - (Managed Opus Only) 'Max Opus Bandwidth': 0
  - (Managed G.729 Only) 'Jitter Compensation': **Enable**
  - (Managed G.729 Only) 'RTCP Mode': **Generate Always**
  - (Managed G.729 Only) 'Dynamic Jitter Buffer Minimum Delay': 40
  - (Managed G.729 Only) 'Jitter Buffer Max Delay': 500
- (Managed G.729 Only) MPLSMode (ini file parameter): 0 if no MPLS; 1 if operating in an MPLS environment

- **Smart Transcoding:** If the SBC (Enterprise or Access SBC) detects WAN network impairments during a call between the Enterprise SBC and Access SBC, the SBC employs voice transcoding by switching the coder from G.711 to Opus for that specific call only. Transcoding is done only on the path between the Enterprise SBC and Access SBC. As Smart Transcoding is applied only on a per call basis, it preserves valuable DSP resources that may be required for other functionalities. An advantage of using the Opus coder is that it consumes less bandwidth than G.711 and overcomes packet loss (by dynamic packet redundancy), allowing the SBC to support more concurrent calls than with G.711 for the same bandwidth. This mode is useful for WAN networks that are relatively stable, allowing the use of G.711 whenever possible and switching to Opus only during adverse network conditions.



Configuration of the Enterprise SBC:

- Device's License Key includes the SBC transcoding feature
- Coder Groups table:
  - Coders Group with G.711
  - Coders Group with Opus
- Allowed Audio Coders Groups table:
  - Allowed Audio Coders Group with G.711
  - Allowed Audio Coders Group with Opus
- IP Profiles table - main IP Profile:
  - 'Extension Coders Group': Select the Coders Group with G.711
  - 'Allowed Audio Coders': Select the Allowed Audio Coders Group with G.711
  - 'Allowed Coders Mode': **Restriction**
  - 'RTCP Feedback': **Feedback On**
  - 'Voice Quality Enhancement': **Enable**
- IP Profiles table - alternative IP Profile:
  - 'Extension Coders Group': Select the Coders Group with Opus
  - 'Allowed Audio Coders': Select the Allowed Audio Coders Group with Opus
  - 'Allowed Coders Mode': **Restriction**
  - 'RTP Redundancy Mode': **Enable**
  - 'RTCP Feedback': **Feedback On**
  - 'Voice Quality Enhancement': **Enable**
  - 'Max Opus Bandwidth': 80000
- Quality of Service Rules table (see Configuring Quality of Service Rules):
  - 'Rule Metric': **Poor InVoice Quality**
  - 'Alternative IP Profile Name': name of Alternative IP Profile (above)

Configuration of the Access SBC for both Smart Transcoding and Managed Opus is listed below. (For Managed G.729, configuration is the same as the Enterprise SBC.)

- Coder Groups table:
  - Coders Group with G.711 and Opus
  - Coders Group with Opus
- Allowed Audio Coders Group table: Allowed Audio Coders Group with Opus
- IP Profiles table - main IP Profile:
  - 'Extension Coders Group': Select the Coders Group with G.711 and Opus
  - 'Voice Quality Enhancement': **Enable**
  - 'RTP Redundancy Mode': **Enable**
  - 'RTCP Feedback': **Feedback On**
  - 'Max Opus Bandwidth': 0
- IP Profiles table - alternative IP Profile:
  - 'Extension Coders Group': Select the Coders Group with Opus
  - 'Allowed Audio Coders': Select the Allowed Audio Coders Group with Opus
  - 'Allowed Coders Mode': **Restriction**
  - 'Voice Quality Enhancement': **Enable**
  - 'RTP Redundancy Mode': **Enable**
  - 'RTCP Feedback': **Feedback On**
  - 'Max Opus Bandwidth': 0
- Quality of Service Rules table (see Configuring Quality of Service Rules):
  - 'Rule Metric': **Poor InVoice Quality**
  - 'Alternative IP Profile Name': name of Alternative IP Profile (above)

- VoIPerfect is applicable only to G.711 and G.729 calls.
- If you are deploying a third-party device between the Enterprise SBC and Access SBC, make sure that the third-party device adheres to the following:
  - ✓ Enable RFC 2198 in SDP negotiation
  - ✓ Enable TMMBR in SDP negotiation
  - ✓ Forward the SDP with feedback (SAVPF) as is
  - ✓ Forward TMMBR messages as is
  - ✓ Forward RTCP messages as is (not terminate them)
  - ✓ (Smart Transcoding only) Forward re-INVITE messages for using Opus as is
  - ✓ (Smart Transcoding only) Forward the SIP header, X-Ac-Action as is

Limiting SBC Call Duration

You can configure the maximum allowed call duration (in minutes) per SBC call. If an established call reaches this user-defined limit, the device terminates the call. The feature ensures that calls are properly terminated, allowing available resources for new calls. The following procedure describes how to configure the feature for all calls (globally). To configure the feature per specific calls, use IP Profiles (IpProfile_SBCMaxCallDuration).

➢ **To configure maximum call duration:**

1. Open the SBC General Settings page (**Setup** menu > **Signaling & Media** tab > **SBC** folder > **SBC General Settings**).
2. In the 'Max Call Duration' field (SBCMaxCallDuration), enter the maximum call duration per SBC call:

3. Click **Apply**.

Playing Tone upon Call Connect

You can configure the device to play a specific tone (recorded audio message / announcement) from a loaded PRT file upon call connection (after SIP 200 OK). The tone can be played to both called and calling parties. When the device finishes playing the tone, the call is connected and the call parties can begin talking.

This feature is configured using a Message Manipulation rule that contains the variable `var.call.src|dst.PlayToneOnConnect`, which specifies the recorded tone to play from the PRT file. The rule is then assigned to the call party (IP Group) to which you want the device to play the tone.

If the device fails to play the tone for whatever reason (for example, the PRT file is not loaded or the specified tone index doesn't exist in the file), you can configure the device to connect or disconnect the call.

➢ **To configure play of tone upon call connect:**

1. Record your tone (.wav file) and convert it to a loadable PRT file, using AudioCodes DConvert utility (see Call Progress Tones File on page 865). The tone must be defined in DConvert as an **acUserDefineTone**<Index> tone type (e.g., acUserDefineTone50).
2. Load the PRT file to the device (see Loading Auxiliary Files on page 862).
3. In the Message Manipulations table (see Configuring SIP Message Manipulation on page 634), configure a rule to specify the tone (index) you recorded in Step 1 and the call party (source or destination) you want it played to. Below is an example for configuring the device to play the tone to call source and destination:

- 'Index': 0 (plays to called party)
  - 'Manipulation Set ID': **1**
  - 'Message Type': **invite.request**
  - 'Condition': **Header.From contains '100'**
  - 'Action Subject': **var.call.dst.PlayToneOnConnect**
  - 'Action Type': **Add**
  - 'Action Value': **'50'**
- 'Index': 1 (plays to calling party)
  - 'Manipulation Set ID': **1**
  - 'Message Type': **invite.request**
  - 'Condition': **Header.From contains '100'**
  - 'Action Subject': **var.call.src.PlayToneOnConnect**
  - 'Action Type': **Add**
  - 'Action Value': **'50'**

4. In the IP Groups table, assign the Manipulation Set ID that you configured in Step 3 to the relevant IP Group (see Configuring IP Groups).
5. Configure what the device should do if it can't play the tone:
   a. Open the SBC General Settings page (**Setup** menu > **Signaling & Media** tab > **SBC** folder > **SBC General Settings**).
   b. From the 'Play Tone on Connect Failure Behavior' drop-down list, select one of the following:
      - **Disconnect** - disconnects the call
      - **Ignore** - connects the call

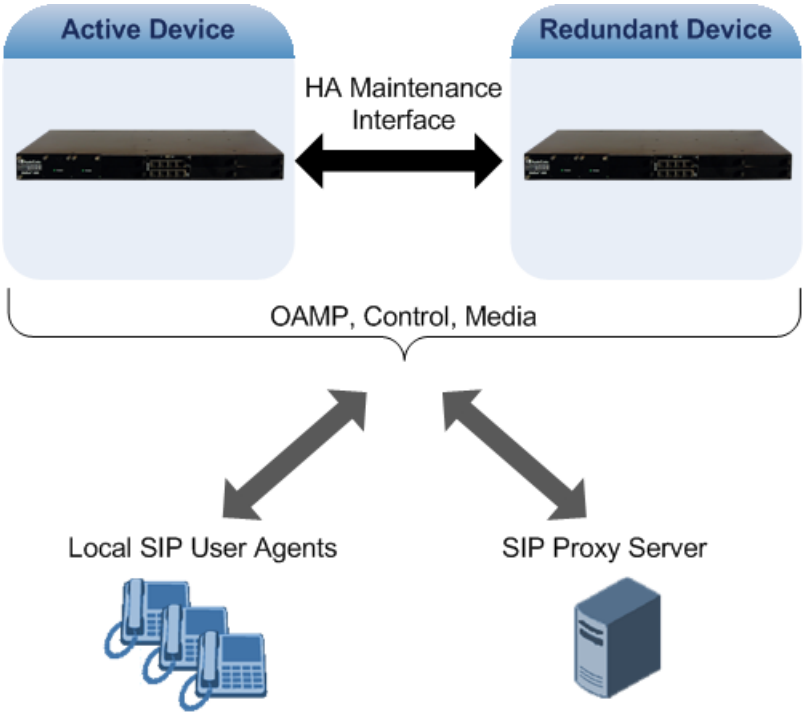
Part VI

High-Availability System

29 HA Overview

The device's High Availability (HA) feature provides 1+1 system redundancy using two devices. If failure occurs in the active device, a switchover occurs to the redundant device which takes over the call handling process. Thus the continuity of call services is ensured. All active calls (signaling and media) are maintained upon switchover.

The figure below illustrates the Active- Redundant HA devices under normal operation. Communication between the two devices is through a Maintenance interface, having a unique IP address for each device. The devices have identical software and configuration including network interfaces (i.e., OAMP, Control, and Media), and have identical local-port cabling of these interfaces.



Connectivity and Synchronization between Devices

In HA mode, the Ethernet connectivity between the two devices is through a special LAN interface on each device, referred to as the *Maintenance* interface. Each device has its own Maintenance interface with a unique address, and each device knows the Maintenance address of the other. The Maintenance interface can use a dedicated Ethernet port group or share the same Ethernet port group with the other network interface types (i.e., OAMP, Media, and Control).

When only one of the devices is operational it is in HA stand-alone state. This means that the device has no connectivity to the second device. When the second device is powered up, it recognizes the active device through the Maintenance network and acquires the HA redundant state. It then begins synchronizing for HA with the active device through the Maintenance network. During synchronization, the active device sends the redundant device its current

configuration settings, including Auxiliary files. The active device also sends its software file (.cmp) if the redundant device is running a different software version. Once loaded to the redundant device, the redundant device reboots to apply the new configuration and/or software. This ensures that the two units are synchronized regarding configuration and software.

Thus, under normal operation, one of the devices is in active state while the other is in redundant state, where both devices share the same configuration and software. Any subsequent configuration update or software upgrade on the active device is also done on the redundant device.

In the active device, all logical interfaces (i.e., Media, Control, OAMP, and Maintenance) are active. In the redundant device, only the Maintenance interface is active, which is used for connectivity to the active device. Therefore, management is done only through the active device. Upon a failure in the active device, the redundant device becomes active and activates all its logical interfaces exactly as was used on the active device.

- If the active device runs an earlier version (e.g., 7.0) than the redundant device (e.g., 7.2), the redundant device is downgraded to the same version as the active device (e.g., 7.0).
- You can delay the transition from HA non-operational state, which occurs during HA synchronization between the active and redundant device, to HA operational state. This is configured by the [HAOperationalStateDelayInSec] parameter. This may be useful, for example, to delay HA switchover when using switches with spanning tree protocol (STP) that take a long time until their ports (to which the redundant device is connected) is ready.

Device Switchover upon Failure

When a failure occurs in the active device, a switchover occurs to the redundant device making it the new active device. Whether a switchover is later done back to the repaired failed device, depends on whether you have enabled the Preempt mode:

- **Enabled:** The Preempt mode specifies one of the device's as the "preferred" device. This is done by assigning different priority levels (1 to 10, where 1 is the lowest) to the two devices. Typically, you would configure the active device with a higher priority level (number) than the redundant device. The only factor that influences the configuration is which device has the greater number; the actual number is not important. For example, configuring the active with 5 and redundant with 4, or active with 9 and redundant with 2 both assign highest priority to the active device. Whenever the device with higher priority recovers from a failure, it first becomes the redundant device but then initiates a switchover to become the active device once again; otherwise, after recovery, it becomes the redundant device and remains as redundant. If you change the priority level of the redundant device to one that is higher than the active device and then reset the redundant device, a switchover occurs to the redundant device making it the active device and the "preferred" device. If both devices are configured with the same priority level, Preempt mode is disabled. Please see note below when using priority level 10.

- **Disabled:** A switchover is done only upon failure of the currently active device.

Failure detection by the devices is done by the constant keep-alive messages they send between themselves to verify connectivity. Upon detection of a failure in one of the devices, the following occurs:

- **Failure in active device:** The redundant device initiates a switchover. The failed device resets and the previously redundant device becomes the active device in stand-alone mode. If at a later stage this newly active device detects that the failed device has been repaired, the system returns to HA mode. If Preempt mode is enabled and the originally active device was configured with a higher priority, a switchover occurs to this device; otherwise, if it was configured with a lower priority (or Preempt mode was disabled), the repaired device is initialized as the redundant device.
- **Failure in redundant device:** The active device moves itself into stand-alone mode until the redundant device is returned to operation. If the failure in the redundant device is repaired after reset, it's initialized as the redundant device once again and the system returns to HA mode.

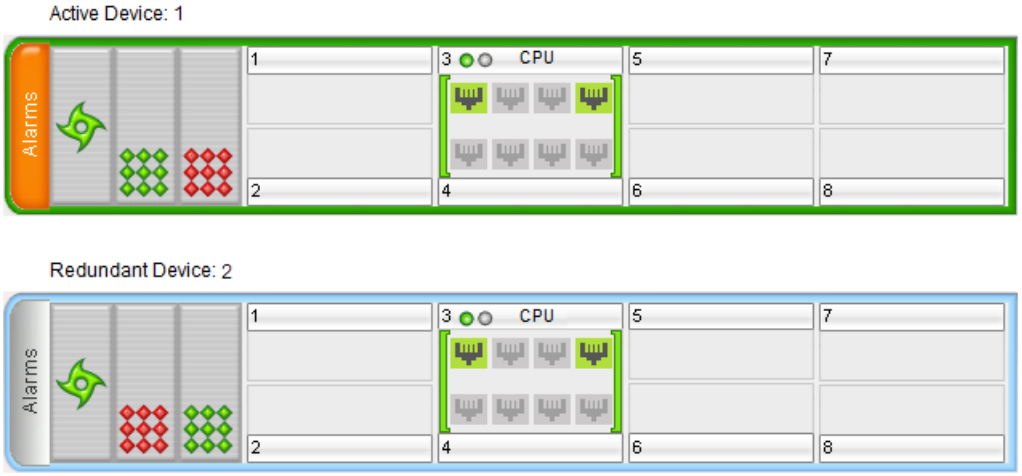
Connectivity failure triggering a switchover can include, for example, one of the following:

- **Loss of physical (link) connectivity:** If one or more physical network groups (i.e., Ethernet port pair) used for one or more network interfaces of the active device disconnects (i.e., no link) and these physical network groups are connected OK on the redundant device, a switchover occurs to the redundant device.
- **Loss of network (logical) connectivity:** No network connectivity, verified by keep-alive packets between the devices. This applies only to the Maintenance interface.

- Switchover triggered by loss of physical connectivity in one or more Ethernet Group is not done if the active device has been configured to a Preempt mode level of 10. In such a scenario, the device remains active.
- After HA switchover, the active device updates other hosts in the network about the new mapping of its Layer-2 hardware address to the global IP address, by sending a broadcast gratuitous Address Resolution Protocol (ARP) message.

Viewing HA Status on Monitor Web Page

You can view the status of the HA system on the Monitor page of the device's Web interface. The page provides a graphical display of both active and redundant devices, as shown below:



You can distinguish between active and redundant devices as follows:

- **Active device:**
  - Color of border surrounding device is green.
  - Title above device is "Active Device". The default name is "Device 1".
- **Redundant device:**
  - Color of border surrounding device is blue.
  - Title above device is "Redundant Device". The default name is "Device 2".

The Monitor page also displays the HA operational status of the device to which you are currently logged in. This is displayed in the 'HA Status' field under the Device Information pane:

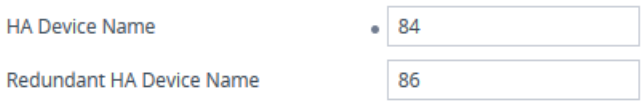
- "Synchronizing": Redundant device is synchronizing with Active device
- "Operational": The device is in HA mode
- "Stand Alone": HA is configured, but the Redundant device is missing and HA is currently unavailable

To view active alarms raised by the Active and Redundant devices, click the **Alarms** area to open the Active Alarms page (see Viewing Active Alarms on page 953).

You can change the name of each device, as described in the following procedure:

➢ To define a name for the device:

1. Open the HA Settings page (**Setup** menu > **IP Network** tab > **Core Entities** folder > **HA Settings**).
2. In the 'HA Device Name' field, enter a name for the active device.
3. In the 'Redundant HA Device Name' field, enter a name for the redundant device.



4. Click **Apply**.

- Once the devices are running in HA mode, you can change the name of the redundant device through the active device only, in the 'Redundant HA Device Name' field.
- When the device sends alarms to OVOC, the name is displayed at the beginning of the alarm description in OVOC, for example, "(SBCSITE01): Proxy lost. looking for another proxy". However, the name is not displayed when OVOC **initially** connects to the device and retrieves the alarms (from the device's Active Alarms table).

30 HA Configuration

This section describes HA configuration.

Prerequisites for HA

The following lists the prerequisites for the HA feature:

- The HA feature is available only if both devices participating in the HA system are installed with a License Key that includes this feature. For installing a License Key, see License Key.
- The physical connections of the first and second devices to the network (i.e., Maintenance interface and OAMP, Control and Media interfaces) **must be identical**. This also means that the two devices must use the same Ethernet Groups and the port numbers belonging to these Ethernet Groups. For example, if the first device uses Ethernet Group 1 (with ports 1 and 2), the second device must also use Ethernet Group 1 (with ports 1 and 2).
- Each network interface of the first device must be connected to the same broadcast domain as the identical network interface on the second device. For example, both Management interfaces must be connected to the same broadcast domain, and so on.
- Before configuring HA, determine the required network topology, as described in Network Topology Types and Rx/Tx Ethernet Port Group Settings.
- Maintenance interfaces restrictions:
  - The Maintenance interface on each device must be configured (each with a different IP address).
  - The maximum delay allowed between the Maintenance interfaces of both devices is 50 ms.
  - The Maintenance interface should be able to perform a fast switchover in case of link failure. Therefore, Spanning Tree Protocol (STP) should not be used in this network. The Ethernet connectivity of the Maintenance interface between the two devices should be constantly reliable without any disturbances.

Initial HA Configuration

By default, HA is disabled on the device. When a device is loaded with valid HA configuration and is the first device to be loaded, it becomes the active device. The second device that is loaded with HA configuration becomes the redundant (standby) device.

Network Topology Types and Rx/Tx Ethernet Port Group Settings

Initial HA configuration depends on how you want to deploy your HA system in the network. The Maintenance interface, which is used for the HA link between Active and Redundant units, can use a dedicated Ethernet Device and Ethernet Group, or share the same Ethernet Device and Ethernet Group with other IP network interface types (such as OAMP, Media and Control).

However, it is recommended that you configure the Maintenance interface with a dedicated Ethernet Device and Ethernet Group (port) to separate it from other IP network interfaces

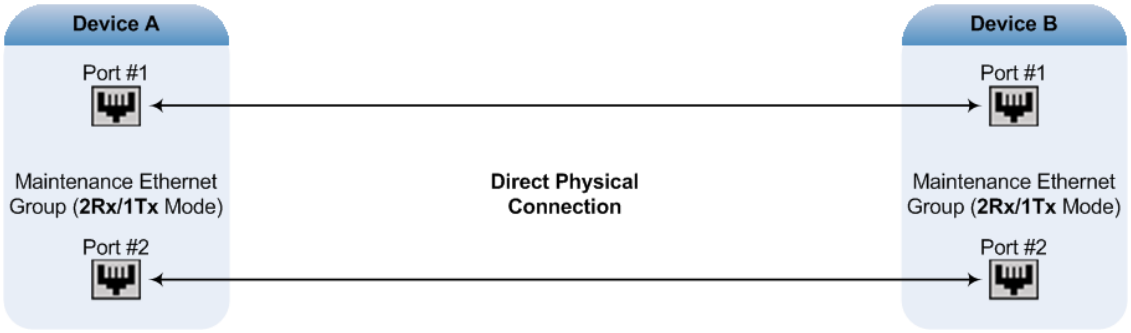
If you want to separate the Maintenance interface from other interfaces, the separation must also be done externally to the units, either physically (different physical networks) or logically (using VLANs). When using VLANs, make sure that you use a different Ethernet Device for each IP network interface (see Configuring Underlying Ethernet Devices and Configuring IP Network Interfaces).

- The Maintenance interface is used for heartbeats and data transfer from active to standby device and therefore, any short interval interruption in communication may cause undesired switchovers.
- If you assign the same Underlying Ethernet Device to all the IP network interfaces, logical separation of traffic may not occur.

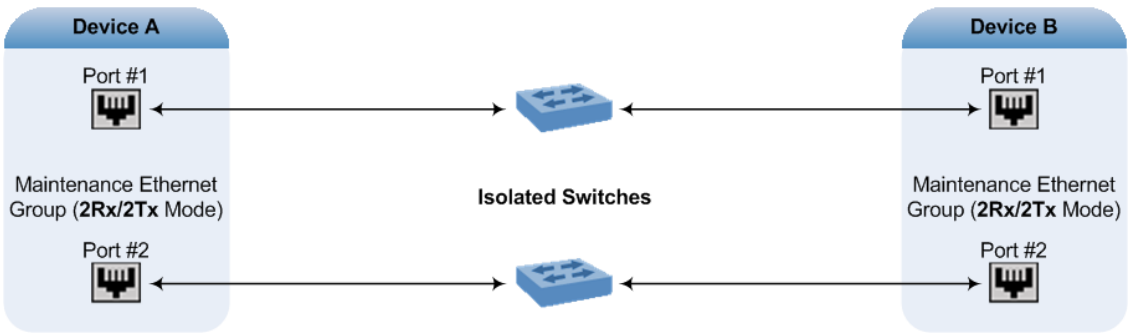
The Maintenance interface can employ Ethernet port redundancy (recommended), by using two ports. This is enabled by configuring the Ethernet Group associated with the Maintenance interface with two ports.

The required receive (Rx) and transmit (TX) mode for the port pair in the Ethernet Group used by the Maintenance interface is as follows:

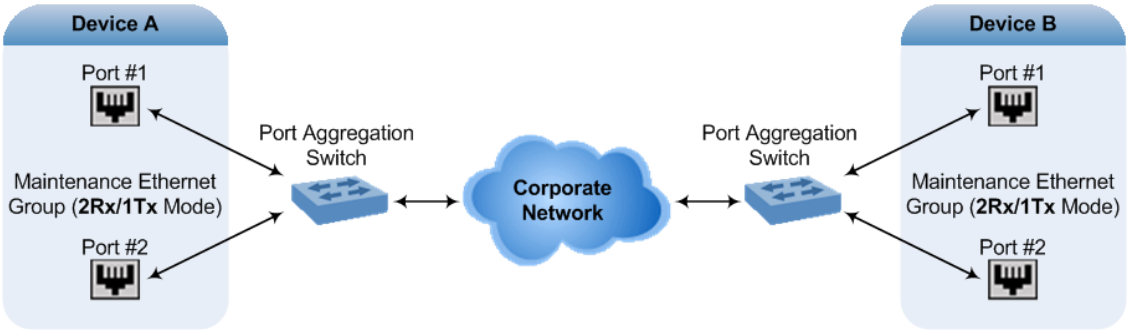
- (Recommended Physical Connectivity) If the Maintenance ports of both devices are connected directly to each other without intermediation of switches, configure the mode to **2RX/1TX**:



- If the two devices are connected through two (or more) isolated LAN switches (i.e., packets from one switch cannot traverse the second switch), configure the mode to **2RX/2TX**:



- For Geographical HA (both units are located far from each other), **2Rx/1Tx** port mode connected to a port aggregation switch is the recommended option:



- When two LAN switches are used, the LAN switches must be in the same subnet (i.e., broadcast domain).
- To configure Rx/Tx modes of the Ethernet ports, see Configuring Ethernet Port Groups

Configuring the HA Devices

To initially configure the two devices comprising the HA system:

1. Configure the first device for HA (see Step 1: Configure the First Device)
2. Configure the second device for HA (see Step 2: Configure the Second Device)
3. Activate HA on the devices (see Step 3: Initialize HA on the Devices)

Step 1: Configure the First Device

The first stage is to configure the first device for HA.

During this stage, make sure that the second device is powered off or disconnected from the network.

➢ **To configure the first device for HA:**

1. Configure the network interfaces, including the default OAMP interface:
   a. Connect your PC to the device using a local, direct physical cable connection and then access the Web interface using the default OAMP network address. For more information on initial access, see Assigning the OAMP IP Address.
   b. Open the IP Interfaces table (see Configuring IP Network Interfaces).
   c. Change the default OAMP network settings to suit your networking scheme.
   d. Configure the Control and Media network interfaces, as required.
   e. Add the HA Maintenance interface (i.e., the **MAINTENANCE** Application Type).

> **Make sure that the Maintenance interface uses an Ethernet Device and Ethernet Group that is not used by any other IP network interface. The Ethernet Group is associated with the Ethernet Device, which is assigned to the interface.**

The IP Interfaces table below shows an example where the Maintenance interface is configured with Ethernet Device "vlan 2" (which is associated with Ethernet Group "GROUP_2"), while the other interface is assigned "vlan 1" (associated with Ethernet Group "GROUP_1"):

INDEX	NAME	APPLICATION TYPE	INTERFACE MODE	IP ADDRESS	PREFIX LENGTH	DEFAULT GATEWAY	PRIMARY DNS	SECONDARY DNS	ETHERNET DEVICE
0	Voice	OAMP + Media +	IPv4 Manual	10.8.40.47	16	10.8.0.1	0.0.0.0	0.0.0.0	vlan 1
1	maint	MAINTENANCE	IPv4 Manual	10.3.0.11	16	10.3.0.1	0.0.0.0	0.0.0.0	vlan 2

2. If the connection is through a switch, the packets of both interfaces should generally be untagged. To do this, open the Ethernet Devices table (see Configuring Underlying Ethernet Devices), and then configure the 'Tagging' parameter to **Untagged** for the Ethernet Device assigned to the Maintenance interface. The figure below shows an example (highlighted) where VLAN 2 is configured as the Native (untagged) VLAN ID of the Ethernet Group "GROUP_2":

INDEX	VLAN ID	UNDERLYING INTERFACE	NAME	TAGGING
0	1	GROUP_1	vlan 1	Untagged
1	2	GROUP_2	vlan 2	Untagged

3. Set the Ethernet port Tx / Rx mode of the Ethernet Group used by the Maintenance interface (see Configuring Ethernet Port Groups). The port mode depends on the type of Maintenance connection between the devices, as described in Network Topology Types and Rx/Tx Ethernet Port Group Settings.
4. Configure HA parameters:
   a. Open the HA Settings page (**Setup** menu > **IP Network** tab > **Core Entities** folder > **HA Settings**):

   b. In the 'HA Remote Address' field, enter the Maintenance IP address of the **second** device.

c. (Optional) Enable the HA Preempt feature by configuring the 'Preempt Mode' parameter to **Enable**, and then setting the priority level of the device in the 'Preempt Priority' field. Make sure that you configure different priority levels for the two devices. Typically, you would configure the active device with a higher priority level (number) than the redundant device. The only factor that influences the configuration is which device has the greater number; the actual number is not important. For example, configuring the active with 5 and redundant with 4, or active with 9 and redundant with 2 both assign highest priority to the active device. Configuring the level to 10 does not cause a switchover upon Ethernet connectivity loss. For more information on the feature, see Device Switchover upon Failure.

5. Burn the configuration to flash **without** a reset.
6. Power down the device.
7. Configure the second device (see Step 2: Configure the Second Device).

Step 2: Configure the Second Device

Once you have configured the first device for HA, you can configure the second device for HA. As the configuration of the second device is similar to the first device, the following procedure briefly describes each step. For detailed configuration such as the path to the Web configuration pages, refer to the section on configuring the first device (Step 1: Configure the First Device).

During this stage, ensure that the first device is powered off or disconnected from the network.

➢ **To configure the second device for HA:**

1. Connect to the device in the same way as you did with the first device.
2. Configure the **same** OAMP, Media, and Control interfaces as configured for the first device.
3. Configure a Maintenance interface for this device. The IP address must be different to that configured for the Maintenance interface of the first device. The Maintenance interfaces of the devices must be in the same subnet.
4. Configure the **same** Ethernet Groups and VLAN IDs of the network interfaces as configured for the first device.
5. Configure the **same** Ethernet port Tx / Rx mode of the Ethernet Group used by the Maintenance interface as configured for the first device.
6. Configure HA parameters in the HA Settings page:
   a. In the 'HA Remote Address' field, enter the Maintenance IP address of the **first** device.
   b. (Optional) Enable the HA Preempt feature by configuring the 'Preempt Mode' parameter to **Enable**, and then setting the priority level of the device in the 'Preempt Priority' field. Make sure that you configure different priority levels for the two devices.

Typically, you would configure the active device with a higher priority level (number) than the redundant device. The only factor that influences the configuration is which device has the greater number; the actual number is not important. For example, configuring the active with 5 and redundant with 4, or active with 9 and redundant with 2 both assign highest priority to the active device. Configuring the level to 10 does not cause a switchover upon Ethernet connectivity loss. For more information on the HA Preempt feature, see Device Switchover upon Failure.

7. Burn the configuration to flash **without** a reset.
8. Power down the device.
9. Continue to Step 3: Initialize HA on the Devices.

Step 3: Initialize HA on the Devices

Once you have configured both devices for HA as described in the previous sections, follow the procedure below to complete and initialize HA so that the devices become operational in HA. This last stage applies to both devices.

➢ **To initialize the devices for HA:**

1. Cable the devices to the network.

You must connect both ports (two) in the Ethernet Group of the Maintenance interface to the network (i.e., two network cables are used). This provides 1+1 Maintenance port redundancy.

2. Power up the devices; the redundant device synchronizes with the active device and updates its configuration according to the active device. The synchronization status is indicated as follows:
   - Active device: The Web interface's Monitor page displays "Synchronizing" in the 'HA Status' field.
   - Redundant device: The ⇆ LED on the E-SBC module is lit yellow.

   When synchronization completes, the redundant device resets to apply the received configuration and software.
   When both devices become operational in HA, the HA status is indicated as follows:
   - Both devices: The Web interface's Monitor page displays "Operational" in the 'HA Status' field.
   - Active device: The ⇆ LED on the E-SBC module is lit green.
   - Redundant device: The ⇆ LED on the E-SBC module flashes yellow.
3. Access the active device with its' OAMP IP address and configure the device as required. For information on configuration done after HA is operational, see Configuration while HA is Operational.

Quick-and-Easy Initial HA Configuration

You can quickly set up two stand-alone devices into an HA system, by loading the same ini configuration file with special parameters to each device. This ini file contains basic configuration that identifies the devices by MAC address and applies the configuration accordingly.

➢ **To quickly set up two devices for HA:**

1. Make sure that the License Key installed on both devices includes the HA license (see Viewing the License Key on page 871).
2. Obtain the MAC address of each device. The MAC address is displayed on the Device Information page (see Viewing Device Information on page 945).
3. Prepare a single ini file with the following configuration:
   - Parameters relating to the first device ("local"):
     - HALocalMAC - specifies the MAC address
     - (Optional) HAUnitIdName - defines a descriptive name
     - (Optional) HARevertiveEnabled - enables the HA Preempt feature
     - (Optional) HAPriority - defines the priority of the device if the HA Preempt feature is enabled
   - Parameters relating to the second device ("remote"):
     - HARemoteMAC - specifies the MAC address
     - HARemoteAddress - defines the HA Maintenance interface address
     - (Optional) HARemoteUnitIdName - defines a descriptive name
     - (Optional) HARemotePriority - defines the priority of the device if the HA Preempt feature is enabled
   - InterfaceTable - defines the IP Interfaces table with two network interfaces for the following Application Types:
     - OAMP + Media + Control: This is the same for both devices.
     - HA Maintenance: This is the Maintenance address of the device whose MAC address is the value of the HALocalMAC parameter.

   An example of a configured ini file is shown below:

```
HALocalMAC = '00908f737a89'
HAUnitIdName = 'Device_1'
HARevertiveEnabled = 1
HAPriority = 6
HARemoteMAC = '00908f594667'
```

```
HARemoteAddress = 1.1.1.2
HARemoteUnitIdName = 'Device_2'
HARemotePriority = 4

[ InterfaceTable ]
FORMAT InterfaceTable_Index = InterfaceTable_ApplicationTypes,
InterfaceTable_InterfaceMode, InterfaceTable_IPAddress, InterfaceTable_
PrefixLength, InterfaceTable_Gateway, InterfaceTable_InterfaceName,
InterfaceTable_PrimaryDNSServerIPAddress, InterfaceTable_
SecondaryDNSServerIPAddress, InterfaceTable_UnderlyingDevice;
InterfaceTable 0 = 6, 10, 10.33.45.40, 16, 10.33.0.1, "Voice", 0.0.0.0, 0.0.0.0,
"vlan 1";
InterfaceTable 1 = 99, 10, 1.1.1.1, 24, 0.0.0.0, "HA", 0.0.0.0, 0.0.0.0, "vlan 2";
[ \InterfaceTable ]
```

4. Load the file to both devices (see Loading an ini File to the Device on page 98).
5. To test your HA system, perform an HA switchover (see Initiating an HA Switchover on page 844).

- If the HA Preempt feature is enabled, the device with the highest priority becomes the active unit. If the HA Preempt feature is not enabled, the first device to load the file becomes the active unit, or if both load the file simultaneously, the device with the "highest" IP address becomes the active unit.
- When configuration is applied to the device whose MAC is the value of the HARemoteMAC parameter, all HA configuration is swapped between local and remote parameters, including the IP address of the Maintenance interface, which is swapped with the address configured for the HARemoteAddress parameter.

Configuration while HA is Operational

When the devices are operating in HA state, subsequent configuration is as follows:

- All configuration, including HA, is done on the active device **only**.
- Non-HA configuration on the active device is automatically updated on the redundant device (through the Maintenance interface).
- HA-related configuration on the active device is automatically updated on the redundant device:
  - Maintenance interface:
    - Modified Maintenance interface address of the active device: The address is set as the new 'HA Remote Address' value on the redundant device.
    - Modified 'HA Remote Address' value on the active device: The address is set as the new Maintenance interface address on the redundant device (requires a device reset).

- Modifications on all other Maintenance interface parameters (e.g., Default Gateway and VLAN ID): updated to the Maintenance interface on the redundant device.

- 'Preempt Mode' parameter.
- 'Preempt Priority' parameter is set for the active device.
- Modified 'Redundant Preempt Priority' value is set for the redundant device.

If the HA system is already in HA Preempt mode and you want to change the priority of the device, to ensure that system service is maintained and traffic is not disrupted, it is recommended to set the higher priority to the redundant device and then reset it. After it synchronizes with the active device, it initiates a switchover and becomes the new active device (the former active device resets and becomes the new redundant device).

Configuring Firewall Allowed Rules

The Active device communicates with the Redundant device through the Maintenance interface. This interface is used by the Active device for HA maintenance with the Redundant device, for example, synchronizing configuration and software updates and performing an HA switchover (see Connectivity and Synchronization between Devices on page 823 for more information).

HA-maintenance communication uses the following ports:

- UDP ports 669, 670 and 680 - for HA synchronization and keep alive
- TCP ports 2442 and 80 - for HA control and data

Therefore, it is crucial that the network in which your HA system is deployed allows this HA traffic through the Maintenance interface. From Version 7.20A.258.119 and later, the device's internal firewall keeps the above listed ports open, by default. If you have an external firewall located between the Active and Redundant devices, make sure that it keeps these ports open as well (i.e., allows this traffic).

For software versions earlier than 7.20A.258.119 only, if you are configuring firewall rules (see Configuring Firewall Rules) that block specific network traffic, you must also configure firewall rules to allow traffic needed in your deployment. This includes basic traffic (e.g., OAMP, SIP signalling and media) and HA maintenance traffic. The following table shows the firewall rules that you need to add in the Firewall table of the Active device to allow HA traffic. The HA-related firewall rules are from Index 10 through 17, where the following IP addresses are used as an example:

- 10.31.4.61 for the Maintenance interface ("HA_IF") of the Redundant device
- 10.31.4.62 for the Maintenance interface ("HA_IF") of the Active device

Table 30-1: Allowed Firewall Rules for HA for Versions Earlier than 7.20A.258.119

Index	Source IP	Source Port	Prefix Length	Start Port	End Port	Protocol	Use Specific Interface	Interface Name	Action Upon Match	Packet Size	Byte Rate	Byte Burst
0												
...	Various rules for basic traffic											
9												
10	10.31.4.61	669	32	669	669	udp	Enable	HA_IF	Allow	0	0	0
11	10.31.4.62	669	32	669	669	udp	Enable	HA_IF	Allow	0	0	0
12	10.31.4.61	0	32	2442	2442	tcp	Enable	HA_IF	Allow	0	0	0
13	10.31.4.62	0	32	2442	2442	tcp	Enable	HA_IF	Allow	0	0	0
14	10.31.4.61	80	32	0	65535	tcp	Enable	HA_IF	Allow	0	0	0
15	10.31.4.62	80	32	0	65535	tcp	Enable	HA_IF	Allow	0	0	0
16	10.31.4.61	670	32	680	680	udp	Enable	HA_IF	Allow	0	0	0
17	10.31.4.62	670	32	680	680	udp	Enable	HA_IF	Allow	0	0	0
18	0.0.0.0	0	0	0	65535	Any	Disable	--	Block	0	0	0

- The index numbers in the table above may change according to your specific allow and block rules.
- The last rule (Index 18) is an example of a blocking traffic rule (blocks all other traffic).
- Configure the firewall on the Active device. This configuration is automatically applied to the Redundant device.
- If you have an external firewall located between the Active and the Redundant HA Maintenance interfaces, you must open (allow) the same port ranges as configured in the table above, on that external firewall.
- If the device needs to communicate with AudioCodes OVOC, you must also add rules to allow incoming traffic from OVOC. For more information, see Configuring Firewall Rules to Allow Incoming OVOC Traffic on page 184.

Configuring DiffServ for HA Maintenance Traffic

You can configure Differentiated Services (DiffServ) for HA Maintenance traffic which flows between the active and redundant devices on the HA Maintenance network interface. The default DiffServ value for this traffic is 46, which should be sufficient in most setups. However, you may need to increase the DiffServ priority if you are also running non-HA traffic on this network. Prioritizing HA Maintenance traffic ensures low latency for this critical network traffic.

➢ **To configure DiffServ for HA Maintenance traffic:**

1. Load an ini file to the device with the [HAMaintenanceIFDiffServValue] parameter configured to the required DiffServ value - 0 is the lowest priority (best effort) and 63 is the highest priority.
2. Reset the device with a burn-to-flash for your settings to take effect.

Monitoring IP Entities and HA Switchover upon Ping Failure

The device's HA Network Monitor feature monitors the connectivity (*reachability*) with network entities (destinations) by pinging them using Internet Control Message Protocol (ICMP) Echo messages. The feature can be used, for example, to check connectivity with nearby routers (or first hops) that the device uses to reach other destinations for sending calls.

The HA Network Monitor table lets you configure up to 10 monitored rows, where each row can include up to 5 destinations to monitor, defined by hostname (or FQDN), or IP address. You can then configure the device to perform an HA switchover if a user-defined number of monitored rows whose destinations fail to reply to the device's sent pings (i.e., *unreachable* destinations) is reached or exceeded.

- The HA Network Monitor feature is a license-based feature (which is part of the general HA license) and is available only if it is included in the License Key that is installed on the device.
- Switchover decisions of the HA Network Monitor feature are non-functional under the following conditions:
  - ✓ HA is disabled (i.e., active device is in standalone mode).
  - ✓ The HA Preempt Priority feature is used (enabled by the 'Preempt Mode' and 'Preempt Priority' parameters).
  - ✓ The number of Ethernet Groups (Ethernet links) on the redundant device that are in "up" status are less than on the active device.
- Destinations that have never replied to the device's pings are not used to determine reachability status and the unreachability threshold for triggering a switchover. They need to reply at least once to the device's pings in order to participate in the device's logic for this feature.
- Once a switchover occurs, the device does not perform switchover loops due to continued ping failures with the monitored row(s). Once a switchover occurs, the device changes the status of the monitored row(s) to "Reachability Unverified". A second switchover occurs only if the row(s) become reachable again and then unreachable.
- The following SNMP alarms are related to the HA Network Monitor feature:
  - ✓ acHASystemFaultAlarm: This alarm is sent to indicate that you have configured the HA Network Monitor feature, but switchover decisions are non-functional (see above).
  - ✓ acHANetworkMonitorAlarm: This alarm is sent to indicate that all destinations of a specific row in the HA Network Monitor table that replied in the past to the device's pings are now "unreachable".
  - ✓ acHASystemSwitchOverAlarm: This alarm is sent to indicate that an HA switchover has occurred due to the HA Network Monitor feature.

The following procedure describes how to configure monitored network entities through the Web interface. You can also configure it through ini file [HaNetworkMonitor] or CLI (`configure network > high-availability network-monitor`).

➢ **To enable and configure monitoring of network entities using pings:**

1. Open the HA Settings page (**Setup** menu > **IP Network** tab > **Core Entities** folder > **HA Settings**), and then do the following:

   a. From the 'HA Network Monitor' (HAPingEnabled) drop-down list, select **Enable**:

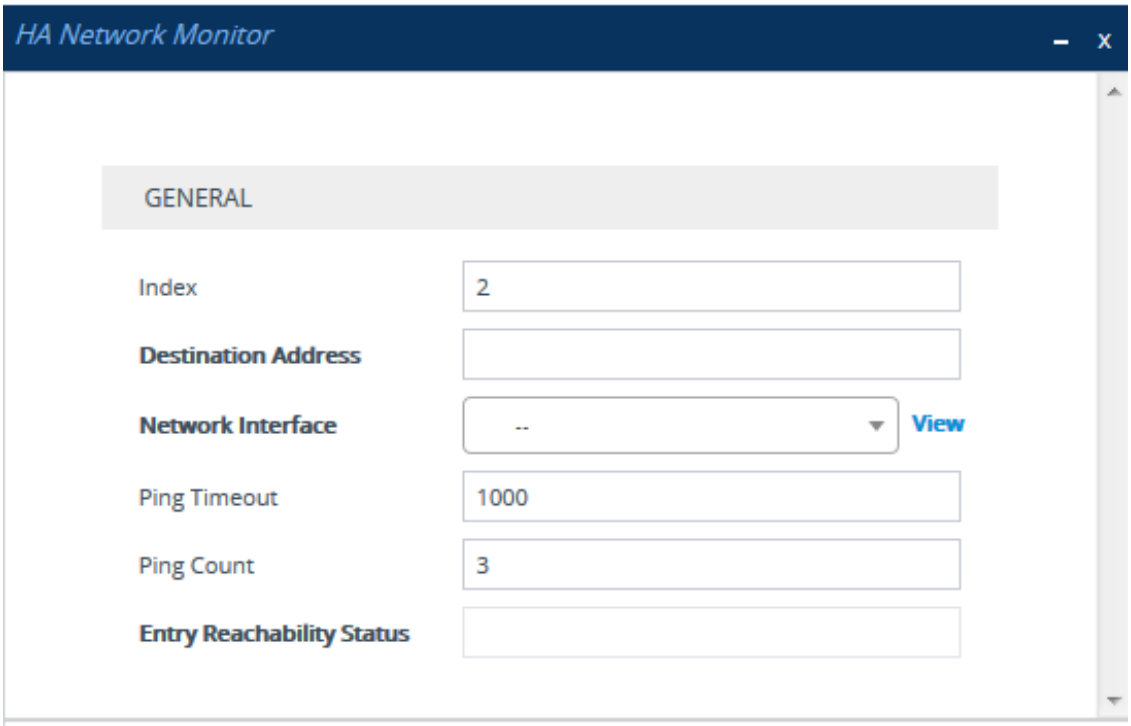
   

   b. In the 'Monitor Threshold' (HaNetworkMonitorThreshold) field, enter the minimum number of failed ("Not Reachable") monitored rows that are required to trigger an HA switchover:

2. Open the HA Network Monitor table (**Setup** menu > **IP Network** tab > **Core Entities** folder > **HA Network Monitor**), and then do the following:

   a. Click **New**; the following dialog box appears:



   b. Configure an HA network monitor entry according to the parameters described in the table below.

   c. Click **Apply**.

Parameter	Description
'Index' [HaNetworkMonitor_Index]	Defines an index number for the new table row. **Note:** Each row must be configured with a unique index.
'Destination Address' `dest-address` [HaNetworkMonitor_DestAddress]	Defines destination addresses of network hosts that you want monitored by the device for the monitored row. The valid value is a hostname (or FQDN) or an IP addresses in dotted-decimal notation. You can configure only one hostname (which can be resolved by DNS into up to five IP addresses). You can configure up to five IP addresses in dotted-decimal notation, where each IP address is separated by a comma or space, for example, "10.1.1.1 20.2.2.2,30.3.3.3" (without quotation marks). **Note:** You must configure the parameter with either a hostname or an IP address (not both).

Parameter	Description
'Network Interface' network-interface [HaNetworkMonitor_NetworkInterface]	Assigns one of the device's IP network interface from where you want to send ping requests to the monitored destinations configured for the monitored row. To configure IP interfaces, see Configuring IP Network Interfaces on page 124.
'Ping Timeout' ping-timeout [HaNetworkMonitor_PingTimeout]	Defines how often (in milliseconds) the device sends ping requests to the destinations configured for the monitored row. This also provides the device time to wait for a reply (if any) from the destination. For example, if configured to 100, the device pings the destination every 100 ms. If the device receives a reply from a destination within this timeout, it considers the destination as online (reachable). If no reply has been received from a user-defined number of consecutive pings (see the 'Ping Count' parameter, below), the device considers the destination as offline (unreachable). The valid value is 100 to 60000. The default is 1000.
'Ping Count' ping-count [HaNetworkMonitor_PingCount]	Defines the number of consecutive failed pings (no replies) before the device considers the destination as offline (unreachable). For example, if you configure the parameter to 2, the destination is considered unreachable after 2 consecutive pings evoked no reply. If this destination later replies to any subsequent ping, the device considers it reachable. The valid value is 1 to 10. The default is 3. **Note:** If the destination has never replied to a ping, the device does not consider it unreachable. Instead, it considers it as undetermined ("Reachability Unverified").
'Entry Reachability Status'	Read-only field displaying the connectivity (reachable) status with the monitored row, which is based on ping results of all its configured destinations: ■ "Reachability Unverified": The reachability status of the monitored row is currently undetermined. In other words, **all** the destinations configured for the monitored row have never replied to the device's pings. ■ "Reachable": The device considers the monitored row as online (reachable). In other words, the device has received a ping reply from **at least one** of the destinations configured for the monitored row.

Parameter	Description
	■ "Not Reachable": The device considers the monitored row as offline (unreachable). In other words, the number of failed pings equals to (or is greater than) that configured by the 'Ping Count' parameter, for **all** the destinations configured for the monitored row and on condition that all these destinations have replied in the past to the device's pings. The status of the monitored row returns to "Reachable" if at least one of the destinations replies to a ping. ■ "Host not resolved": The hostname (if configured in the 'Destination Address' parameter) couldn't be resolved into an IP address.

Once you have configured the destinations to monitor, you can view the status of each destination of a selected monitored row, as described in the following procedure.

➢ **To view reachability status of destinations per monitored row:**

1. Open the HA Network Monitor table (**Setup** menu > **IP Network** tab > **Core Entities** folder > **HA Network Monitor**).
2. In the table, select the required index row (monitored row), and then click the **HA Network Monitor Peers Status** link located below the table; the HA Network Monitor Peers Status table appears, displaying the reachability status of each destination ('Peer Destination Address') of the monitored row, as shown in the below example:

INDEX	PEER DESTINATION ADDRESS	PEER REACHABILITY STATUS
0	10.4.4.60	Reachable
1	10.4.0.1	Reachable

The reachability status is displayed in the 'Peer Reachability Status' read-only field:

- "Reachability unverified": The reachability status of the destination is currently undetermined. In other words, the destination has never replied to the device's pings.
- "Reachable": The device considers the destination as online (reachable). In other words, the device has received a ping reply from the destination.
- "Not reachable": The device considers the destination as offline (unreachable). In other words, the number of consecutive failed pings equaled to (or was greater than) that configured by the 'Ping Count' parameter.
- "Terminated by ping error": The device is unable to send a ping to the destination (typically, due to a routing issue or incorrect destination address). To resolve the problem, correct your routing configuration or the address of the destination, and then

enter the edit mode of the HA network monitor row belonging to the destination and click **Apply** to refresh your changes.

The 'Ping Loss Percentage' read-only field displays the percentage of pings sent to the destination that failed to get a reply, in the last five minutes.

31 HA Maintenance

This section describes HA maintenance.

Maintenance of Redundant Device

The only interface that is operational on the redundant device is the Maintenance interface. The following protocols can be used for maintenance purposes for this interface:

- **Syslog:** To receive Syslog messages from the redundant device, make sure that you have configured a valid VLAN and routing rule from the Maintenance network to the Syslog server.
- **Telnet:** A Telnet server is always available on the redundant device (even if disabled by configuration).

The active device runs the above maintenance protocols on its' OAMP interface.

Replacing a Failed Device

If you need to replace a faulty device, the new device must be configured exactly as the second device, as described in Configuring the HA Devices.

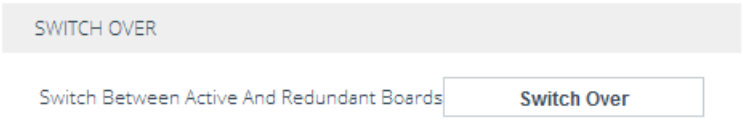
Initiating an HA Switchover

You can initiate a switchover from the Active to Redundant device.

When performing an HA switchover, the HA mode becomes temporarily unavailable.

➢ **To perform a switch-over:**

1. Open the High Availability Maintenance page:
   - Toolbar: Click the **Actions** button, and then from the drop-down menu, choose **Switchover**.
   - Navigation tree: **Setup** menu > **Administration** tab > **Maintenance** folder > **High Availability Maintenance**.



2. Click **Switch Over**; a confirmation box appears requesting you to confirm.
3. Click **OK**.

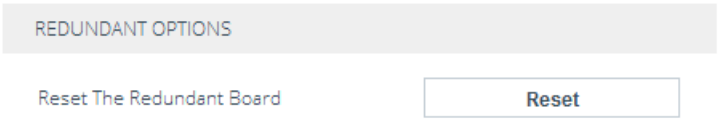
Resetting the Redundant Unit

You can reset the Redundant device, if necessary.

> When resetting the Redundant device, the HA mode becomes temporarily unavailable.

➢ **To reset the Redundant device:**

1. Open the High Availability Maintenance page:
   - Toolbar: Click the **Actions** button, and then from the drop-down menu, choose **Switchover**.
   - Navigation tree: **Setup** menu > **Administration** tab > **Maintenance** folder > **High Availability Maintenance**.



2. Click **Reset**; a confirmation box appears requesting you to confirm.
3. Click **OK**.

Software Upgrade

You can perform the following types of software upgrades on the HA system:

- **Software Upgrade with Device Reset:** Both active and redundant devices burn and reboot with the new software version. This method is quick and simple, but it disrupts traffic (i.e., traffic affecting).
- **Hitless Software Upgrade:** This method maintains service (i.e., not traffic affecting).

For more information, see Software Upgrade.

Disconnecting and Reconnecting HA

You can disconnect the two devices in the HA system and return them to standalone devices.

➢ **To disconnect HA through CLI:**

1. From the CLI of the active device, connect (Telnet) to the redundant device, by typing the following command:

```
# debug ha conn-to-red
```

2. Log in to the redundant device's CLI with its username (e.g., **Admin**) and password:

```
Username: Admin
Password:
```

3. Access the Privileged mode, by typing the following command, and the mode's password:

```
> enable
Password:
#
```

4. At the prompt, type the following command and enter a new OAMP IP address for the redundant device:

```
# debug ha disconnect-system <New OAMP Address for Redundant Device>
```

The redundant device resets, the HA status of the active device changes to "Standalone", and the redundant device now becomes a non-HA device.

- The new OAMP address of the redundant device must be different to the active device.
- The HA Maintenance network interface (in the IP Interfaces table) on the redundant device is unaffected by the command.
- All **Media + Control** network interfaces (in the IP Interfaces table) on the redundant device are deleted.
- The 'HA Remote Address' [HARemoteAddr] field value, which specifies the HA Maintenance address of the active device is deleted on the redundant device.

You can later restore the HA system, by following the below procedure.

➢ **To restore the HA system through CLI:**

1. On the previously **redundant** device, configure the HA Maintenance interface address of the active device:

```
(config-network)# high-availability
(ha)# remote-address <HA Maintenance Address of Active Device>
```

The HA Maintenance address can alternatively be configured in the Redundant device's Web interface's 'HA Remote Address' parameter.

2. Reset the redundant device:

```
(ha)# do reload now
```

The procedure assumes that no network changes were made to both devices' HA Maintenance interface or Ethernet Devices (VLAN); otherwise, the devices may not be able to communicate with each other.

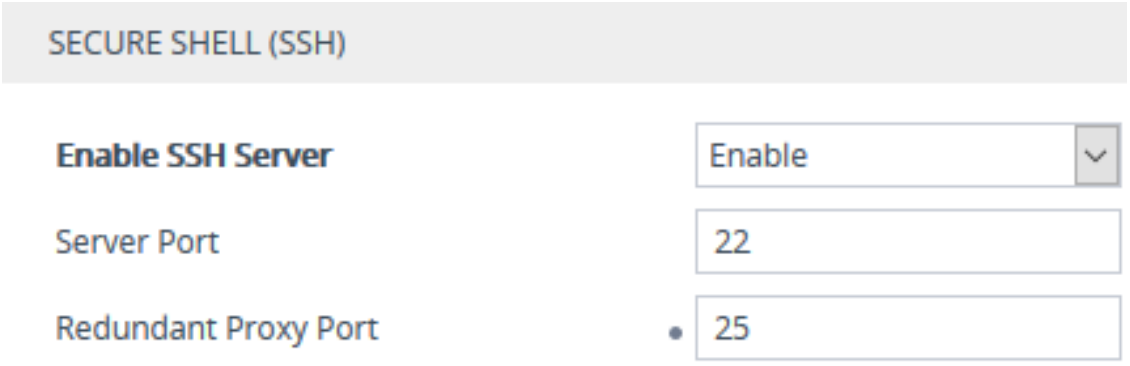
Accessing Files on Redundant Device from Active through SSH

From the active device, you can access files stored on the redundant device's flash memory, through SSH (or SFTP). Once accessed, your SSH client can download files to your PC (e.g., debug file, locally stored CDR file, and Configuration Package file) or rename files (e.g., locally stored CDR file). This eliminates the need to perform an HA switchover to change the redundant device to active in order to access the device's files.

Access to the redundant device uses a proxy SSH server port on the active device. As this is a secure connection, your SSH client needs to authenticate itself with the device (SSH server). The device performs standard user authentication, where you can configure it to check the user's credentials in the device's Local Users table or with a remote authentication server. Access is granted only to users with Security Administrator user level.

➢ **To access redundant device from active device for file download:**

1. Enable SSH (see Enabling SSH with RSA Public Key for CLI on page 70).
2. Open the CLI Settings page (**Setup** menu > **Administration** tab > **Web & CLI** folder > **CLI Settings**).
3. In the 'Redundant Proxy Port' field (SSHRedundantProxyPort), configure a port for SSH through which the active device accesses the redundant device. This proxy port must be different to the active device's regular SSH server port (configured in the 'Server Port' parameter).



4. Click **Apply**.

The device may take a long time to prepare the debug file for transfer when it contains much information. Some SFTP clients (for example, WinSCP and FileZilla) have a short default connection timeout and if the file transfer is not started within this timeout, the transfer attempt is aborted. Therefore, it is recommended to configure a longer timeout for your SFTP client application.

Backing Up and Restoring HA Configuration

Once you have your HA system up and running, you can make a backup of your HA configuration, by saving an ini configuration file from the active device. If your HA system fails, for whatever reason, you can simply load this file to the devices to restore HA.

➢ **To backup and then restore HA:**

1. Save the ini file from the active device to a folder on your PC. Store this file in a safe location for future use (i.e., HA backup).
2. If a failure occurs in one or both devices, load the backup ini file to the failed device(s).

If one or both devices in the HA system have been replaced (RMA) for whatever reason (e.g., a hardware failure), before loading the file, update the file with the new MAC addresses ([HALocalMAC] or [HARemoteMAC] parameters).

Part VII

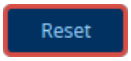
Maintenance

32 Basic Maintenance

This section describes basic maintenance.

Resetting the Device

You can reset the device through the device's management tools. Device reset may be required for maintenance purposes. Certain parameters require a device reset for their settings to take effect. These parameters are displayed in the Web interface with the lightning ⚡ symbol. In addition, whenever you make any configuration change that requires a reset, the **Reset** button on the Web interface's toolbar is displayed with a red border, as shown below:



The Web interface also provides you with the following options when resetting the device:

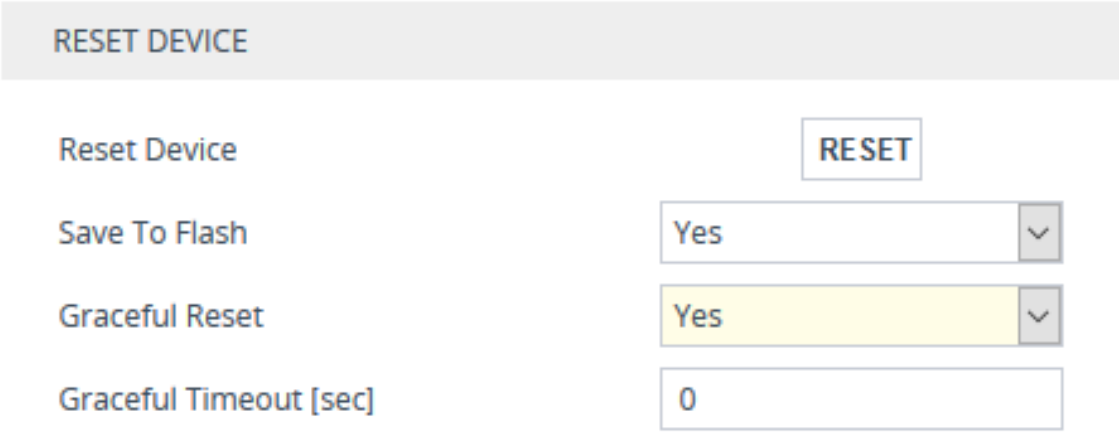
- Save current configuration to the device's flash memory (non-volatile) prior to reset
- Reset the device only after a user-defined time (G*raceful Reset*) to allow current calls to end (calls are terminated after this interval)

To reset the device (and save configuration to flash) through CLI, use the following command:

```
# reset now
```

➢ **To reset the device through Web interface:**

1. Open the Maintenance Actions page:
   - Toolbar: Click the **Reset** button.
   - Navigation tree: **Setup** menu > **Administration** tab > **Maintenance** folder > **Maintenance Actions**.



2. From the 'Save To Flash' drop-down list, select one of the following:
   - **Yes:** Current configuration is saved (*burned*) to flash memory prior to reset (default).

- **No:** The device resets without saving the current configuration to flash. All configuration done after the last configuration save will be discarded (lost) after reset.

3. From the 'Graceful Reset' drop-down list, select one of the following:
   - **Yes:** The device reset only after a user-defined time, configured in the 'Graceful Timeout' field (see next step). During this interval, no new traffic is accepted. If no traffic exists and the time has not yet expired, the device resets immediately.
   - **No:** The device resets immediately, regardless of traffic. Any existing traffic is immediately terminated.
4. In the 'Graceful Timeout' field (available only if you have configured the 'Graceful Reset' field to **Yes**), enter the time (in seconds) after which the device resets. Note that if no traffic exists and the time has not yet expired, the device resets.
5. Click the **Reset** button; a confirmation message box appears, requesting you to confirm.
6. Click **OK** to confirm device reset; if you configured the 'Graceful Reset' field to **Yes** (in Step 3), the reset is delayed and a screen appears displaying the number of remaining calls and time. When the device begins to reset, a message appears to notify you.

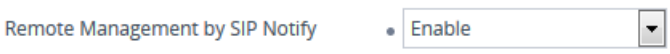
Remotely Resetting Device using SIP NOTIFY

The device can be remotely reset upon the receipt of a SIP NOTIFY message that contains an Event header set to 'check-sync;reboot=true' (proprietary to AudioCodes), as shown in the example below:

```
NOTIFY sip:<user>@<dsthost> SIP/2.0
To: sip:<user>@<dsthost>
From: sip:sipsak@<srchost>
CSeq: 10 NOTIFY
Call-ID: 1234@<srchost>
Event: check-sync;reboot=true
```

➢ **To enable remote reset upon receipt of SIP NOTIFY:**

1. Open the SIP Definitions General Settings page (**Setup** menu > **Signaling & Media** tab > **SIP Definitions** folder > **SIP Definitions General Settings**).
2. From the 'Remote Management by SIP Notify' (EnableSIPRemoteReset) drop-down list, select **Enable**:



3. Click **Apply**.

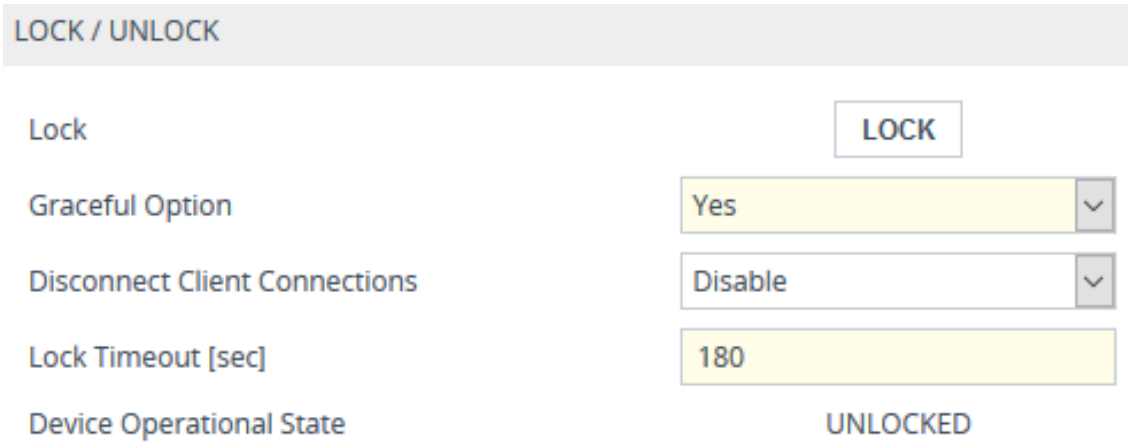
Locking and Unlocking the Device

You can lock the device so that it stops processing calls. This may be useful, for example, when you want to upload new software files to the device and you don't want any traffic to interfere with the process. Locking the device may be done gracefully, whereby the device stops accepting new calls, but allows existing calls to continue for up to a user-defined duration before terminating them.

> You can also configure the device to wait without a timeout until all active calls end on their own, before going into lock state. This is done through the CLI, using the following command: `# admin state lock graceful forever`

➢ **To lock the device:**

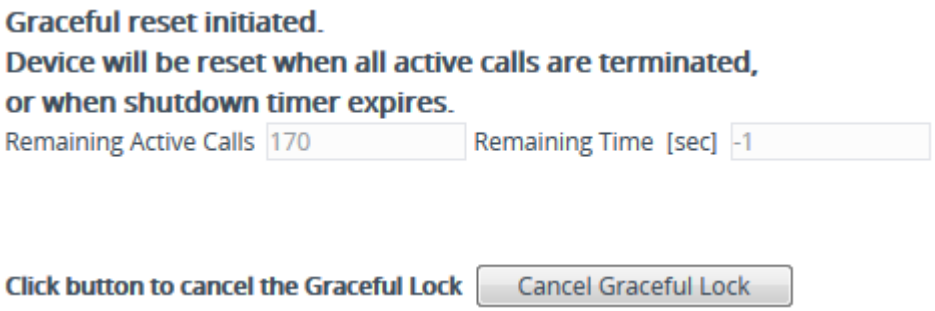
1. Open the Maintenance Actions page:
   - Toolbar: Click the **Reset** button.
   - Navigation tree: **Setup** menu > **Administration** tab > **Maintenance** folder > **Maintenance Actions**.



2. From the 'Graceful Option' drop-down list, select one of the following options:
   - **Yes:** The device locks only after a user-defined duration, configured in the 'Lock Timeout' field (see next step). During this interval, no new traffic is accepted, allowing only existing calls to continue until the timeout expires. If at any time during this timeout there are no active calls, the device locks. If there are still active calls when the timeout expires, the device terminates them and locks.
   - **No:** The device locks immediately, terminating all existing traffic.

   Note: These options are available only if the current status of the device is in "UNLOCKED" state.
3. If you configured 'Graceful Option' to **Yes** (see previous step), then in the 'Lock Timeout' field, enter the time (in seconds) after which the device locks.

4. If you also want the device to terminate (close) existing TLS/TCP client connections and reject new incoming TLS/TCP client connections during the locked state, then from the 'Disconnect Client Connections' drop-down list, select **Enable**. If disabled (default), existing client connections will remain and incoming TLS/TCP client connections will be accepted during the locked state.
5. Click the **LOCK** button; a confirmation message box appears requesting you to confirm device lock.
6. Click **OK** to confirm;
   - If you configured 'Graceful Option' to **Yes**, a lock icon is displayed and a window appears displaying the number of remaining calls and time. To cancel the lock, click the **Cancel Graceful Lock** button.

   - If you configured 'Graceful Option' to **No**, the lock process begins immediately.

   The 'Device Operational State' read-only field displays "LOCKED" and the device does not process any calls.

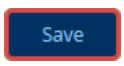
➢ **To unlock the device:**

- Click the **UNLOCK** button; the device unlocks immediately and accepts new incoming calls. The 'Device Operational State' read-only field displays "UNLOCKED".

Saving Configuration

When you configure parameters and tables in the Web interface and then click the **Apply** button on the pages in which the configurations are done, changes are saved to the device's *volatile* memory (RAM). These changes revert to their previous settings if the device subsequently resets (hardware or software) or powers down. Therefore, to ensure that your configuration changes are retained, you must save them to the device's non-volatile memory (i.e., flash memory).

To save your settings to flash, click the **Save** button located on the toolbar. To remind you to save your settings to flash, the **Save** button is displayed with a red border, as shown below:



To save configuration to flash through CLI, use the following command:

```
# write
```

33 Channel Maintenance

This chapter describes channel-related maintenance.

Disconnecting Active Calls

You can forcibly disconnect all active calls, or disconnect specific calls based on Session ID.

➢ **To disconnect calls through CLI:**

- Disconnect all active calls:

```
# clear voip calls
```

- Disconnect active calls belonging to a specified Session ID:

```
# clear voip calls <Session ID>
```

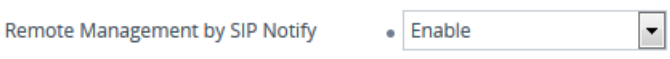
Remotely Disconnecting Calls using SIP NOTIFY

The device can be triggered to disconnect all current calls upon the receipt of a SIP NOTIFY message containing an Event header with the value 'soft-sync' (proprietary to AudioCodes), as shown in the example below:

```
NOTIFY sip:<user>@<dsthost> SIP/2.0
To: sip:<user>@<dsthost>
From: sip:sipsak@<srchost>
CSeq: 10 NOTIFY
Call-ID: 1234@<srchost>
Event: soft-sync
```

➢ **To enable remote call disconnect upon receipt of SIP NOTIFY:**

1. Open the SIP Definitions General Settings page (**Setup** menu > **Signaling & Media** tab > **SIP Definitions** folder > **SIP Definitions General Settings**).
2. From the 'Remote Management by SIP Notify' (EnableSIPRemoteReset) drop-down list, select **Enable**:



3. Click **Apply**.

34 Upgrading the Device's Software

You can use the Web interface's Software Upgrade Wizard to easily upgrade the device's software version (.cmp file). You can also use the wizard to load an *ini* file and Auxiliary files (e.g., CPT file). However, you can only use the wizard if you at least load a .cmp file. Once loaded, you can select other file types to load.

You can also use the wizard to upgrade devices in High Availability (HA) mode. You can choose between two optional HA upgrade methods:

- **System Reset Upgrade (non-Hitless):** Both active and redundant devices are upgraded simultaneously. Therefore, this method is traffic-affecting and terminates current calls during the upgrade process. The process is as follows:
  a. The active (current) devices loads the .cmp file.
  b. The active devices sends the .cmp file to the redundant deviceblade.
  c. Both active and redundant devices install and burn the file to flash memory with a reset. In other words, no HA switchover occurs.
- **Hitless Upgrade:** The devices are upgraded without disrupting traffic (i.e., current calls are maintained). The process is as follows:
  a. The active (current) device loads the .cmp file.
  b. The active device sends the .cmp file to the redundant device.
  c. The redundant device installs and burns the file to its flash memory with a reset. The redundant device now runs the new software version.
  d. An HA switchover occurs from active to redundant deviceblade. Therefore, current calls are maintained and now processed by the previously redundant deviceblade, which is now the active deviceblade.
  e. The previously active device (now in redundant mode) installs and burns the file to flash memory with a reset. Therefore, both devices now run the new software version.
  f. An HA switchover occurs from active device (i.e., the initial redundant device) to redundant device (i.e., the initial active device) to return the devices to their original HA state. Only the initial redundant device undergoes a reset to return to redundant state.

- You can obtain the latest software version files from AudioCodes website (registered users only) at https://www.audiocodes.com/library/firmware.
- When you start the wizard, the rest of the Web interface is unavailable. After the files are successfully installed with a device reset, access to the full Web interface is restored.
- If you upgraded your firmware (.cmp file) and the "SW version mismatch" message appears in the Syslog or Web interface, your License Key does not support the new .cmp file version. If this occurs, contact AudioCodes support team for assistance.
- Currently, hitless software **downgrade** from Version 7.2.150 to an earlier version is not supported (and the non-hitless method must be used).
- To upgrade the device from any earlier version to 7.0, do the following:
  1) Delete core dumps from the redundant device through CLI (Telnet). Core dump deletion can take up to 10 minutes.
  2) Perform a manual switchover from active to redundant.
  3) When the system is operational again, delete core dumps from the current redundant device through CLI (Telnet). Core dump deletion can take up to 10 minutes.
  4) Start the Hitless Software Upgrade procedure.
- Instead of manually upgrading the device, you can use the device's Automatic Update feature for automatic provisioning (see Automatic Provisioning).

The following procedure describes how to load files using the Web interface's Software Upgrade Wizard.

Alternatively, you can load files using the CLI:

- cmp file:

```
copy firmware from <URL>
```

- ini or Auxiliary file:

```
copy <ini file or auxiliary file> from <URL>
```

- CLI Script file:

```
copy cli-script from <URL>
```

- HA devices:
  - Hitless Software Upgrade:

```
# copy firmware from <URL and file name>
```

  - Non-Hitless Software Upgrade:

```
# copy firmware from <URL and file name> non-hitless
```

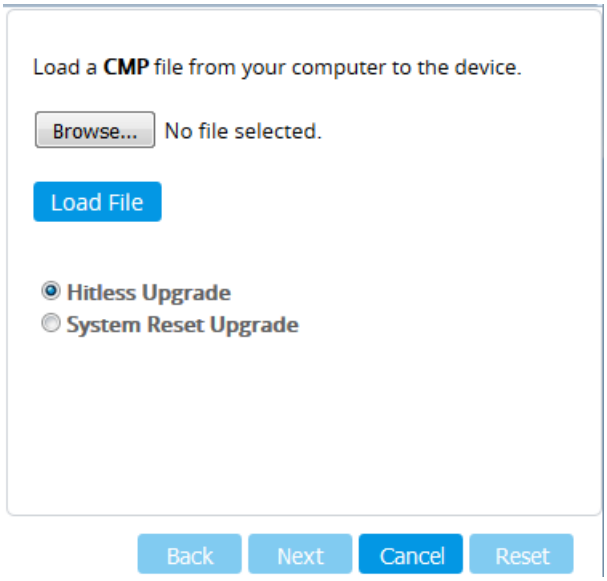
If you load the firmware file through CLI, when you initiate the copy command a message is displayed in the console showing the load progress. If other management users are connected to the device through CLI, the message also appears in their CLI sessions, preventing them from performing further actions on the device and disrupting the upload process. For more information, refer to the CLI Reference Guide.

➢ **To upgrade the device using the Software Upgrade wizard:**

1. Make sure that you have installed a License Key that is compatible with the software version to be installed (see License Key).
2. It is recommended to enable the Graceful Lock feature (see Locking and Unlocking the Device). The wizard resets the device at the end of the upgrade process, thereby causing current calls to be untimely terminated. To minimize traffic disruption, the Graceful Lock feature prevents the establishment of new calls.
3. It is recommended to backup the device's configuration to your computer. If an upgrade failure occurs, you can restore your configuration by uploading the backup file to the device. For more information, see Configuration File.
4. Open the Software Upgrade wizard:
   - **Toolbar:** From the **Actions** drop-down menu, choose **Software Upgrade**.
   - **Navigation tree:Setup** menu > **Administration** tab > **Maintenance** folder > **Software Upgrade**.

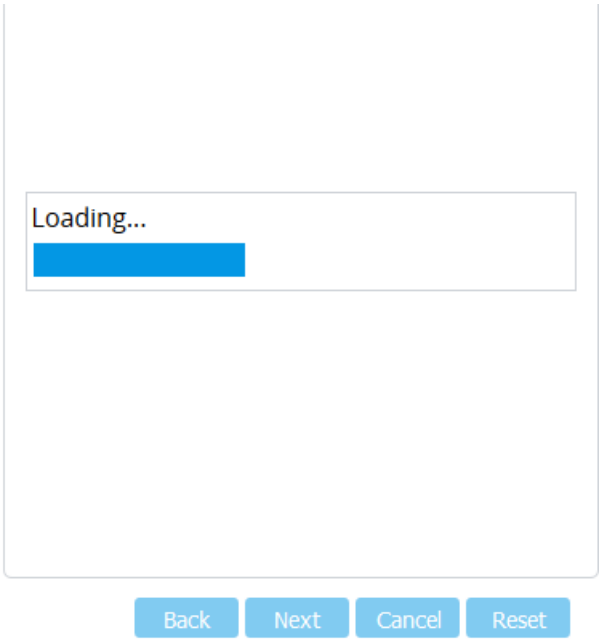
   

5. Click **Start Software Upgrade**; the wizard starts, prompting you to load a .cmp file:



- The Hitless Upgrade and System Reset Upgrade options appear only if the device is configured for HA.
- At this stage, you can quit the Software Upgrade wizard without having to reset the device, by clicking **Cancel**. However, if you continue with the wizard and start loading the .cmp file, the upgrade process must be completed with a device reset.

6. Click **Browse**, and then navigate to and select the .cmp file.
7. Click **Load File**; the device begins to install the .cmp file and a progress bar displays the status of the loading process:



When the file is loaded, a message is displayed to inform you that the file was successfully loaded.

8. If your device is in HA mode, select one of the following upgrade options:

- Hitless Upgrade (default)
- System Reset Upgrade

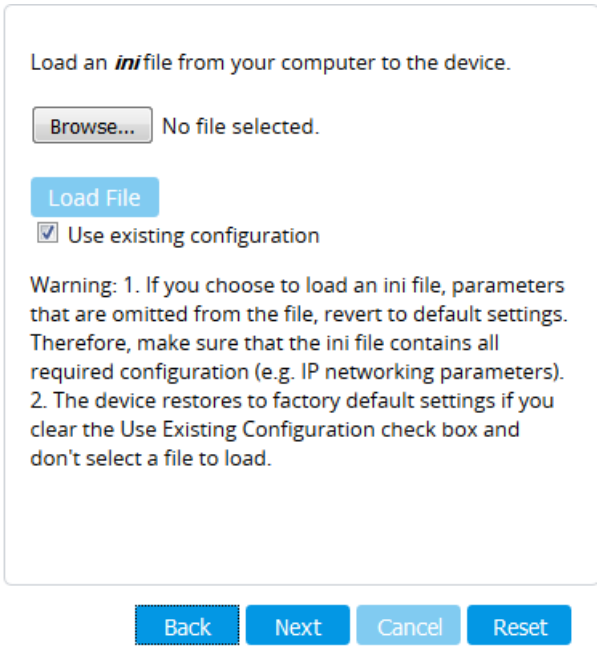
See the description of these methods in the beginning of this section.

> If you select the **Hitless Upgrade** option, the wizard can only be used to upload a .cmp file; Auxiliary and ini files cannot be uploaded.

9. To load additional files, use the **Next** and **Back** buttons to navigate through the wizard to the desired file-load wizard page; otherwise, skip to the next step to load the .cmp file only.

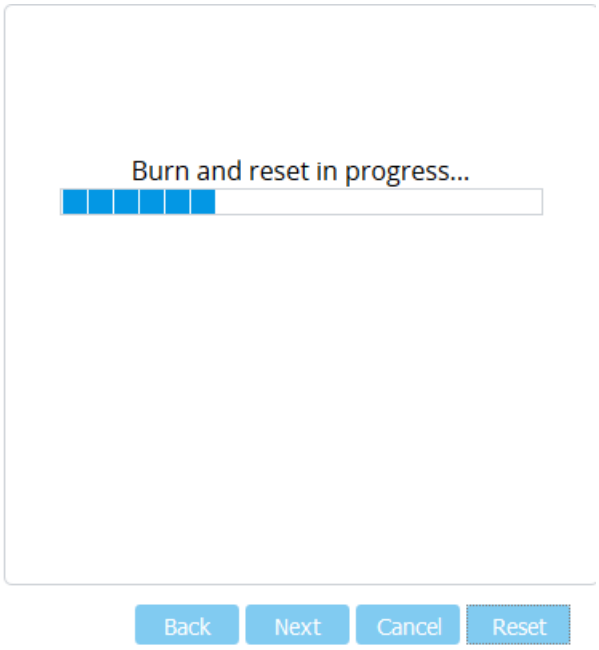
   The wizard page for loading an *ini* file lets you do one of the following:

   - **Load a new ini file:**
     i. Click **Browse**, and then navigate to and select the new ini file.
     ii. Click **Load File**; the device loads the *ini* file.
   - **Restore configuration to factory defaults:** Clear the 'Use existing configuration' check box.
   - **Retain the existing configuration (default):** Select the 'Use existing configuration' check box.



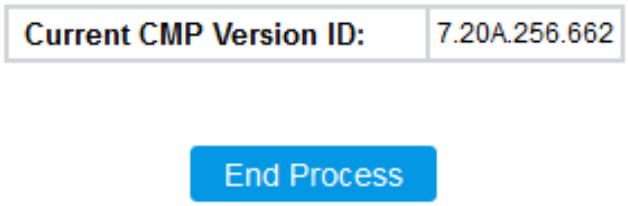
> If you use the wizard to load an *ini* file, parameters excluded from the *ini* file are assigned default values (according to the .cmp file) and thereby, overwrite values previously configured for these parameters.

10. Click **Reset**; a progress bar is displayed, indicating the progress of saving the files to flash and device reset.



Device reset may take a few minutes (even up to 30 minutes), depending on .cmp file version.

After the device finishes the installation process and resets, the wizard displays the End Process page, showing the installed .cmp software version and any other files that you may have also installed. For example:



11. Click **End Process** ; the Web Login screen appears, prompting you to log into the device.
12. Log in with your username and password; a message box appears informing you that the device's software has been upgraded (new .cmp file).
13. Click **OK** to close the message box.

35 Loading Auxiliary Files

You can load Auxiliary files to the device using any of the following methods:

- Web interface - see Loading Auxiliary Files through Web Interface
- CLI - see Loading Auxiliary Files through CLI
- One Voice Operations Center (OVOC) – refer to the *OVOC User's Manual*

You can also automatically load Auxiliary files from a remote server, using the device's Automatic Update mechanism (see Automatic Update Mechanism).

The following table lists the different types of Auxiliary files.

Table 35-1: Auxiliary Files

File	Description
INI	Configures the device. The Web interface enables practically full device provisioning. However, some features may only be configured by ini file or you may wish to configure your device through ini file. For more information, see INI File-Based Management.
Call Progress Tones	Region-specific, telephone exchange-dependent file that contains the Call Progress Tones (CPT) levels and frequencies for the device. The default CPT file is U.S.A. For more information, see Call Progress Tones File.
Prerecorded Tones	The Prerecorded Tones (PRT) file enhances the device's capabilities of playing a wide range of telephone exchange tones that cannot be defined in the CPT file. For more information, see Prerecorded Tones File.
Dial Plan	Provides dialing plans. **Note:** Load a Dial Plan file using the Auxiliary Files page only for **backward compatibility**; otherwise, import Dial Plan files using the Dial Plan table (see Configuring Dial Plans on page 609).
User Info	Loads a User Information file. **Note:** Load a User Information file using the Auxiliary Files page only for **backward compatibility**; otherwise, load the file or configure users in the User Information table for SBC users (see Configuring SBC User Information Table through Web Interface on page 590).
AMD Sensitivity	Answer Machine Detector (AMD) Sensitivity file containing the AMD Sensitivity suites. For more information, see AMD Sensitivity File.

File	Description
SBC Wizard Template Package	Contains the vendor-interoperability configuration templates for the SBC Configuration Wizard. For more information, see SBC Configuration Wizard.

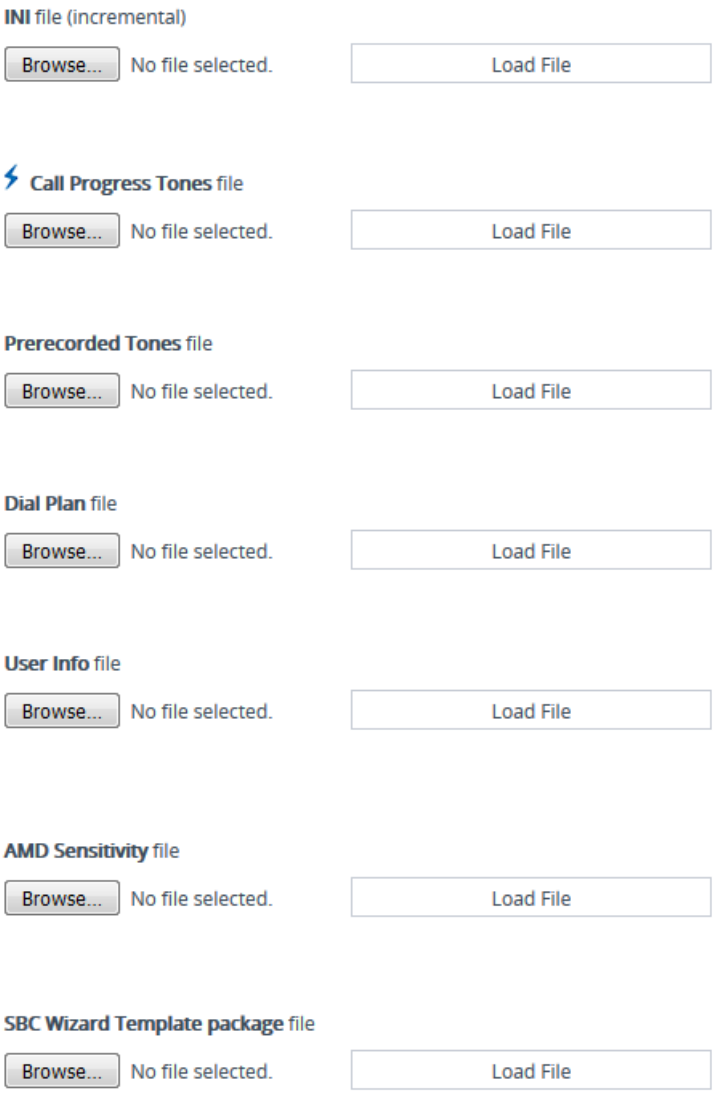
Loading Auxiliary Files through Web Interface

The following procedure describes how to load Auxiliary files through the Web interface.

- When loading an ini file through the Auxiliary Files page, only parameter settings specified in the ini file are applied to the device; all other parameters remain at their current settings.
- If you load an ini file containing Auxiliary file(s), the Auxiliary files specified in the file overwrite the Auxiliary files currently installed on the device.
- For the User Information file, **only** use the Auxiliary Files page for **backward compatibility**. If backward compatibility is not needed, load the file or configure users in the User Information table for SBC users (see Configuring SBC User Information Table through Web Interface on page 590).

➢ **To load Auxiliary files through Web interface:**

1. Open the Auxiliary Files page:
   - Toolbar: From the **Actions** drop-down menu, choose **Auxiliary Files**.
   - Navigation tree: **Setup** menu > **Administration** tab > **Maintenance** folder > **Auxiliary Files**.



2. Click the **Browse** button corresponding to the Auxiliary file type that you want to load, navigate to the folder in which the file is located, and then click **Open**; the name of the file appears next to the **Browse** button.
3. Click the corresponding **Load File** button.
4. Repeat steps 2 through 3 for each file you want to load.
5. If you have loaded a Call Progress Tones file, reset the device with a save-to-flash for your settings to take effect. For all other Auxiliary files, a device reset is not required and you can click the **Save** button instead.

Loading Auxiliary Files through CLI

You can load Auxiliary files from remote servers through CLI:

- **Single Auxiliary file:**

```
# copy <file> from <URL of remote server>
```

For example:

```
# copy call_progress_tones from http://192.169.11.11:80/cpt_us.dat
```

- **Multiple (batch) Auxiliary files:** The Auxiliary files must be contained in a TAR (Tape ARchive) file (.tar). The TAR file can contain any number and type of Auxiliary files (e.g., Dial Plan file and CPT file).

```
# copy aux-package from | to <URL of remote server with TAR file name>
```

For example:

```
# copy aux-package from http://192.169.11.11:80/aux_files.tar
```

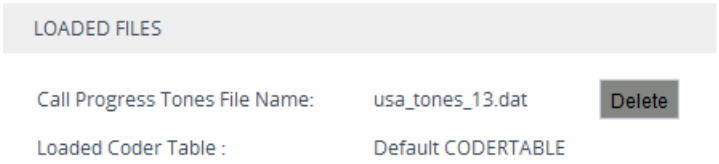
For more information on CLI, refer to the *CLI Reference Guide*.

Deleting Auxiliary Files

You can delete loaded Auxiliary files through the Web interface, as described below.

➢ **To delete a loaded Auxiliary file:**

1. Open the Device Information page (see Viewing Device Information); the loaded files are listed under the Loaded Files group:



2. Click the **Delete** button corresponding to the file that you want deleted; a confirmation message box appears.
3. Click **OK** to confirm.
4. Reset the device with a save-to-flash for your settings to take effect.

Call Progress Tones File

The Call Progress Tones (CPT) Auxiliary file contains definitions of the CPT (levels and frequencies) that are detected and generated by the device.

You can use one of the supplied Auxiliary files (.dat file format) or create your own file. To create your own file, it's recommended to modify the supplied *usa_tone.ini* file (in any standard text editor) to suit your specific requirements and then convert the modified *ini* file into binary *dat* file format, using AudioCodes DConvert utility. For more information, refer to the *DConvert Utility User's Guide*.

The CPT file can only be loaded in .dat file format.

You can create up to 32 different Call Progress Tones, each with frequency and format attributes. The frequency attribute can be single or dual-frequency (in the range of 300 to 1980 Hz) or an Amplitude Modulated (AM). Up to 64 different frequencies are supported. Only eight AM tones, in the range of 1 to 128 kHz, can be configured (the detection range is limited to 1 to 50 kHz). Note that when a tone is composed of a single frequency, the second frequency field must be set to zero.

The format attribute can be one of the following:

- **Continuous:** A steady non-interrupted sound (e.g., a dial tone). Only the 'First Signal On time' should be specified. All other on and off periods must be set to zero. In this case, the parameter specifies the detection period. For example, if it equals 300, the tone is detected after 3 seconds (300 x 10 msec). The minimum detection time is 100 msec.
- **Cadence:** A repeating sequence of on and off sounds. Up to four different sets of on/off periods can be specified.
- **Burst:** A single sound followed by silence. Only the 'First Signal On time' and 'First Signal Off time' should be specified. All other on and off periods must be set to zero. The burst tone is detected after the off time is completed.

You can specify several tones of the same type. These additional tones are used only for tone detection. Generation of a specific tone conforms to the first definition of the specific tone. For example, you can define an additional dial tone by appending the second dial tone's definition lines to the first tone definition in the *ini* file. The device reports dial tone detection if either of the two tones is detected.

The Call Progress Tones section of the *ini* file comprises the following segments:

- **[NUMBER OF CALL PROGRESS TONES]:** Contains the following key:
  'Number of Call Progress Tones' defining the number of Call Progress Tones that are defined in the file.
- **[CALL PROGRESS TONE #X]:** containing the Xth tone definition, starting from 0 and not exceeding the number of Call Progress Tones less 1 defined in the first section (e.g., if 10 tones, then it is 0 to 9), using the following keys:
  - **Tone Type:** Call Progress Tone types:
    - **[1]** Dial Tone
    - **[2]** Ringback Tone
    - **[3]** Busy Tone
    - **[4]** Congestion Tone
    - **[6]** Warning Tone
    - **[7]** Reorder Tone

  - [8] Confirmation Tone
  - [9] Call Waiting Tone - heard by called party
  - [15] Stutter Dial Tone
  - [16] Off Hook Warning Tone
  - **[17]** Call Waiting Ringback Tone (heard by the calling party)
  - **[18]** Comfort Tone
  - **[23]** Hold Tone
  - **[46]** Beep Tone
- **Tone Modulation Type:** Amplitude Modulated (1) or regular (0)
- **Tone Form:** The tone's format can be one of the following:
  - Continuous (1)
  - Cadence (2)
  - Burst (3)
- **Low Freq [Hz]:** Frequency (in Hz) of the lower tone component in case of dual frequency tone, or the frequency of the tone in case of single tone. This is not relevant to AM tones.
- **High Freq [Hz:** Frequency (in Hz) of the higher tone component in case of dual frequency tone, or zero (0) in case of single tone (not relevant to AM tones).
- **Low Freq Level [-dBm]:** Generation level 0 dBm to -31 dBm in dBm (not relevant to AM tones).
- **High Freq Level:** Generation level of 0 to -31 dBm. The value should be set to 32 in the case of a single tone (not relevant to AM tones).
- **First Signal On Time [10 msec]:** 'Signal On' period (in 10 msec units) for the first cadence on-off cycle. For continuous tones, the parameter defines the detection period. For burst tones, it defines the tone's duration.
- **First Signal Off Time [10 msec]:** 'Signal Off' period (in 10 msec units) for the first cadence on-off cycle (for cadence tones). For burst tones, the parameter defines the off time required after the burst tone ends and the tone detection is reported. For continuous tones, the parameter is ignored.
- **Second Signal On Time [10 msec]:** 'Signal On' period (in 10 msec units) for the second cadence on-off cycle. Can be omitted if there isn't a second cadence.
- **Second Signal Off Time [10 msec]:** 'Signal Off' period (in 10 msec units) for the second cadence on-off cycle. Can be omitted if there isn't a second cadence.
- **Third Signal On Time [10 msec]:** 'Signal On' period (in 10 msec units) for the third cadence on-off cycle. Can be omitted if there isn't a third cadence.

- **Third Signal Off Time [10 msec]:** 'Signal Off' period (in 10 msec units) for the third cadence on-off cycle. Can be omitted if there isn't a third cadence.
- **Fourth Signal On Time [10 msec]:** 'Signal On' period (in 10 msec units) for the fourth cadence on-off cycle. Can be omitted if there isn't a fourth cadence.
- **Fourth Signal Off Time [10 msec]:** 'Signal Off' period (in 10 msec units) for the fourth cadence on-off cycle. Can be omitted if there isn't a fourth cadence.
- **Carrier Freq [Hz]:** Frequency of the carrier signal for AM tones.
- **Modulation Freq [Hz]:** Frequency of the modulated signal for AM tones (valid range from 1 to 128 Hz).
- **Signal Level [-dBm]:** Level of the tone for AM tones.
- **AM Factor [steps of 0.02]:** Amplitude modulation factor (valid range from 1 to 50). Recommended values from 10 to 25.

- When the same frequency is used for a continuous tone and a cadence tone, the 'Signal On Time' parameter of the continuous tone must have a value that is greater than the 'Signal On Time' parameter of the cadence tone. Otherwise, the continuous tone is detected instead of the cadence tone.
- The tones frequency must differ by at least 40 Hz between defined tones.

Below shows an example of a configured dial tone to 440 Hz only:

```
[NUMBER OF CALL PROGRESS TONES]
Number of Call Progress Tones=1
#Dial Tone
[CALL PROGRESS TONE #0]
Tone Type=1
Tone Form =1 (continuous)
Low Freq [Hz]=440
High Freq [Hz]=0
Low Freq Level [-dBm]=10 (-10 dBm)
High Freq Level [-dBm]=32 (use 32 only if a single tone is required)
First Signal On Time [10msec]=300; the dial tone is detected after 3 sec
First Signal Off Time [10msec]=0
Second Signal On Time [10msec]=0
Second Signal Off Time [10msec]=0
```

Prerecorded Tones File

The Prerecorded Tone (PRT) file contains up to 80 (and maximum of 10 minutes) user-defined prerecorded tones that can be played by the device. For example, it can be used to play a held tone (music on hold) to a call party that has been put on hold or a ringback tone to a calling party. The PRT file overcomes the limitations of the CPT file (such as limited number of

predefined tones and limited number of frequency integrations in a single tone). The PRT file also lets you play different held and ringback tones for different groups of users. To do this, configure an IP Profile with the required ringback tone (IPProfile_LocalRingbackTone) and/or held tone (IPProfile_LocalHeldTone) from the PRT file, and then associate the IP Profile with the required IP Group.

- The PRT file only generates (plays) tones; detection of tones is according to the CPT file.
- The PRT file can be up to 4 megabytes in size.
- If the PRT file contains a tone that also exists in the CPT file, the tone in the PRT file is played instead (i.e., overrides the tone in the CPT file).
- For SBC calls, you can define a PRT file with multiple tones for the same tone type, but where each tone is defined with a different coder. If the coder of the tone is the same as that used in the current call, DSPs are not required by the device to play the tone. Therefore, if a tone is defined with a coder that is also used in the call, the device always selects this specific tone. However, if the coders are different, the device uses DSPs to play the appropriate tone from the Call Progress Tones (CPT) file (if the tone and CPT file exist).
- The device requires DSPs for local generation of tones.
- The PRT file supports only the ringback tone and hold tone for SBC calls.

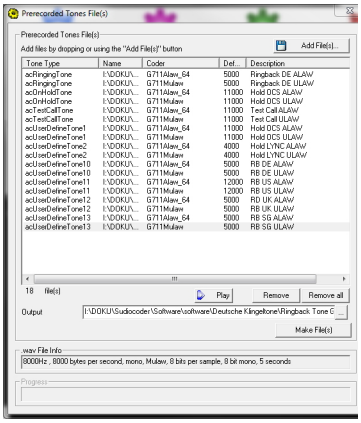
The tones can be recorded using a standard third-party, recording utility (such as Adobe Audition). Once recorded, you need to combine the recorded files into a single and loadable PRT file (.dat), using the latest version of AudioCodes DConvert utility. In DConvert, each recording must be added to the PRT file with the tone type "acUserDefineTone<Index>". When you want to specify the tone (ringback or held tone) to play for a specific IP Profile (IPProfile_LocalRingbackTone and IPProfile_LocalHeldTone parameters), you need to use this index number. For more information on the DConvert utility, refer to the DConvert Utility User's Guide. Once you have created the PRT .dat file, you need to load it to the device (flash memory), using the Web interface (see Loading Auxiliary Files) or CLI.

You must record the tones (raw data files) with the following properties:

- Coders: G.711 A-law, G.711 µ-law, or G.729
- Rate: 8 kHz
- Resolution: 8-bit
- Channels: mono

The PRT file can include prerecorded audio tones of different coders (e.g., some with G.711 and some with G.729). The prerecorded tones are played repeatedly. This allows you to record only part of the tone and then play the tone for the full duration. For example, if a tone has a cadence of 2 seconds on and 4 seconds off, the recorded file should contain only these 6 seconds. The device repeatedly plays this cadence for the configured duration. Similarly, a continuous tone can be played by repeating only part of it.

The following figure shows an example of the creation of the PRT file containing multiple user-defined tones ("acUserDefineTone<Index>") through the DConvert utility:



AMD Sensitivity File

The device is shipped with a default, pre-installed *AMD Sensitivity* file for its Answering Machine Detection (AMD) feature. This file includes the detection algorithms for detecting whether a human or answering machine has answered the call, and is based on North American English. In most cases, the detection algorithms in this file suffice even when your deployment is in a region where a language other than English is spoken. However, if you wish to replace the default file with a different AMD Sensitivity file containing customized detection algorithms, please contact the sales representative of your purchased device for more information.

The AMD Sensitivity file is created in .xml format and then converted to a binary .dat file that can be installed on the device. The XML-to-binary format conversion can be done using AudioCodes DConvert utility. For more information on using this utility, refer to *DConvert Utility User's Guide*.

Only one AMD Sensitivity file can be installed on the device. To install a new AMD Sensitivity file, use one of the following methods:

- Web interface: Auxiliary Files page - see Loading Auxiliary Files.
- TFTP during initialization: Configure the [AMDSensitivityFileName] parameter, and then copy the AMD Sensitivity file to the TFTP directory.
- Automatic Update feature: Configure the [AMDSensitivityFileUrl] parameter through ini file. For more information, see Automatic Update Mechanism.

For more information on the AMD feature, see Answering Machine Detection (AMD).

User Info File

For loading User Info (User Information) files, use the Auxiliary Files page for **backward compatibility only**. If backward compatibility is not needed, load the file or configure users in the User Information table for SBC users (see Configuring SBC User Information Table through Web Interface on page 590). For file syntax when loading a User Information file using the Auxiliary Files page, see the note bulletins in these sections.

36 License Key

The License Key determines the features (e.g., Test Call and voice coders) and various capacity figures (e.g., number of Test Calls and SBC call sessions) that you have ordered for your device.

The local License Key, which is installed on the device through ini file (locally or through the Automatic Update mechanism), contains all the licenses for the ordered features and capacity. However, for the SBC capacity licenses, which includes SBC sessions, transcoding sessions, and registered far-end users, you can use AudioCodes OVOC management tool to provide and manage them. OVOC provides various SBC capacity licensing modes, as described in OVOC-Managed SBC Capacity Licenses on page 881.

- The availability of certain Web pages in the Web interface depends on the licensed features in the License Key.

Viewing the License Key

The License Key is displayed on the License Key page, showing all the device's licensed features and capacity.

➢ **To view License Key through Web interface:**

1. Open the License Key page:
   - **Toolbar:** From the **Actions** drop-down menu, choose **License Key**.
   - **Navigation tree: Setup** menu > **Administration** tab > **License** folder > **License Key**.

- Ordered features are always licensed by the **local** License Key. In other words, even if you are using OVOC to manage the device's SBC capacity licenses, all the other features and capacity figures are licensed by the local License Key.
- If you save the device's ini configuration file to a folder on your computer, the local License Key is also included (see Downloading and Loading ini Configuration File on page 895). If the device is in High-Availability mode, the saved ini file of the Active device also includes the License Key of the Redundant device.

In addition to displaying the licensed features and capacity, the License Key page also displays general information on a bar at the top of the page, as shown in the example below:

	NA	Floating License	5967925	72	Not connected
Product Key	OVOC Product Key	Mode	Serial Number	Device Type	License Server Status

Table 36-1: Description of General information on License Key Page

Field	Description
Product Key	Displays the device's Product Key. For more information, see Viewing the Device's Product Key.
OVOC Product Key	Displays the Product Key of OVOC that is providing and managing the SBC capacity licenses for the device. **Note:** The field only appears when the device uses OVOC to manage its SBC capacity licenses, as described in OVOC-Managed SBC Capacity Licenses on page 881.
Mode	Displays the type of license used for the device's SBC capacity licenses: ■ "Local License Key": The SBC capacity licenses are only based on the local License Key. ■ "License Pool": The SBC capacity licenses are obtained remotely from the Fixed License Pool, which is managed by OVOC. For more information, see Fixed License Pool Model on page 881. ■ "Floating License": The SBC capacity licenses are obtained remotely from the Floating License, which is managed by OVOC. For more information, see Floating License Model on page 884. ■ "Flex License": The SBC capacity licenses are obtained remotely from the Flex License, which is managed by OVOC. For more information, see Flex License Model on page 886.
Serial Number	Displays the device's serial number.
Device Type	Displays AudioCodes internal model identification number of the device.
License Server Status	Displays the connectivity status between the device and OVOC when the device uses OVOC to manage its SBC capacity licenses (Fixed License, Floating License, or Flex License): ■ "Connected": This indicates that the device is connected to OVOC. ■ "Disconnected": This indicates that the device was connected to OVOC, but has lost connection with OVOC due to network problems (HTTPS TCP

Field	Description
	connection). ■ "Not Connected": This indicates that the device has not connected to OVOC. **Note:** The field only appears when the device uses OVOC to manage its SBC capacity licenses, as described in OVOC-Managed SBC Capacity Licenses on page 881.

Local License Key

The local License Key contains all the licenses for your ordered features and capacity. This License Key is installed locally on the device.

- When you install a new License Key, it overwrites the previously installed License Key. Therefore, features that were licensed by the previous License Key and not included in the new License Key will no longer be available.
- The local License Key is unique to the device (based on Serial Number) and cannot be installed on other devices.
- To use OVOC to manage the SBC capacity licenses (SBC sessions, transcoding sessions, and registered users), see OVOC-Managed SBC Capacity Licenses on page 881.
- You can also install the local License Key remotely using the device's Automatic Update mechanism (see Automatic Provisioning on page 901).

Installing License Key through Web Interface

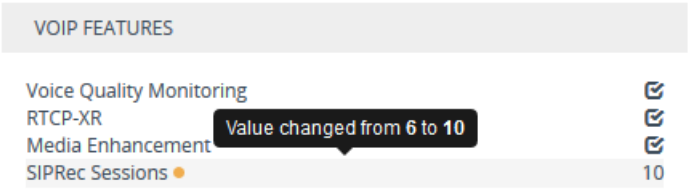
You can install the local License Key through the Web interface using one of the following methods:

- Installing a License Key file (see Installing a License Key File on page 876)
- Installing a License Key string (see Installing a License Key String on the next page)

When you initially load the License Key before installing it, the License Key page uses color-coded icons to indicate the changes between the currently installed License Key and the new License Key that you loaded. The following table describes these color codes:

Table 36-2: Color-Coded Icons for Newly Loaded License Key

Icon	Color	Description
	Green	Indicates new features added by the new License Key.
	Orange	Indicates the capacity change of an existing feature. Move your mouse over the icon to view a pop-up describing the capacity change, as shown in the following example for SIPRec Sessions:

Icon	Color	Description
		
●	Red	Indicates features of the currently installed License Key that are not included in the new License Key and are no longer available.

After you install the License Key (device reset with a save-to-flash), the icons are no longer displayed and the License Key page displays only features and capacity that are licensed by the new License Key.

Installing a License Key String

You can install the License Key as a string in encrypted format through the Web interface.

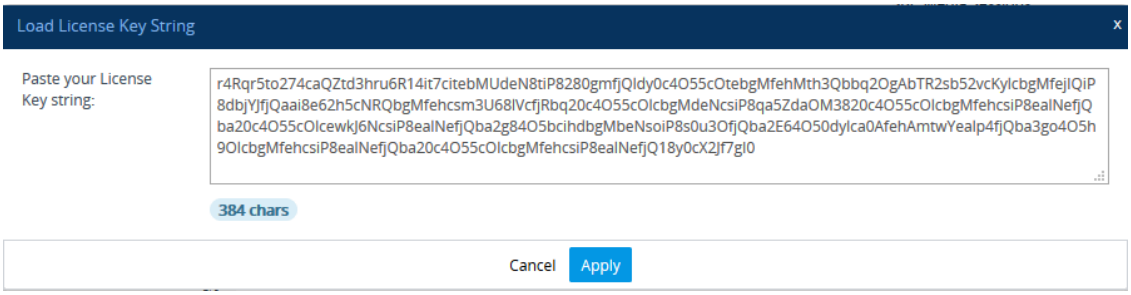
- The License Key installation process includes a device reset and therefore, is traffic-affecting. To minimize disruption of current calls, it is recommended to perform this procedure during periods of low traffic.
- Installation of the License Key in string format is not applicable to devices in HA mode. To install the License Key for devices in HA mode, you need to load a License Key file, as described in Installing on HA Devices on page 877.

➢ To install a License Key string through Web interface:

1. Open the License Key page (see Viewing the License Key on page 871).
2. Back up the currently installed License Key as a precaution. If the new License Key does not comply with your requirements, you can re-load the backed-up License Key to restore the device's original capabilities. For backing up the License Key, see Backing up Local License Key on page 880.
3. Copy the License Key string (from the License Key file or email) to your clipboard. Make sure that you copy **only** the encrypted string (and not the serial number or any other part of the string), as shown in the example below:

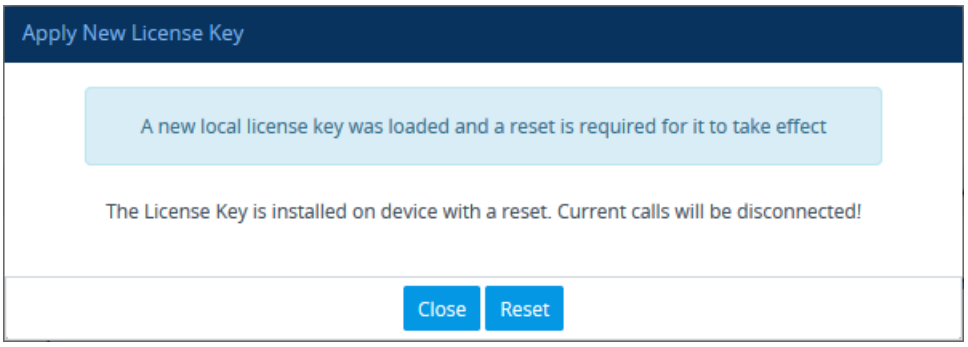
```
[LicenseKeys]
S/N5967925=r4Rqr5to27458ANud3hru6x402R5cOlcbgNfuhcsiP8ealNefjQ9ay0c4O5bcihdbgM8f9GliP8ealNe8OcpdHio4O55cO19bg
Mfehcsjjgealp4820ba28e4O55cOlcbgMfehcsiP8ealtestNefjQba20c4O55cOlcbgMfeggkqN8ealNefjQba20c4O57ciV9bgMD4hcsiP8
eal17pmkba3Ui4O55cOlcbgMfehcsijgcaRVcf3Maai4d4O50dZCMkDfPNhgrh3c19BZanBUba24d4idhcypfQwk62y00
```

4. Click **Load String**; the Load License Key String dialog box appears.
5. In the text box, paste your License Key string, as shown in the following example:

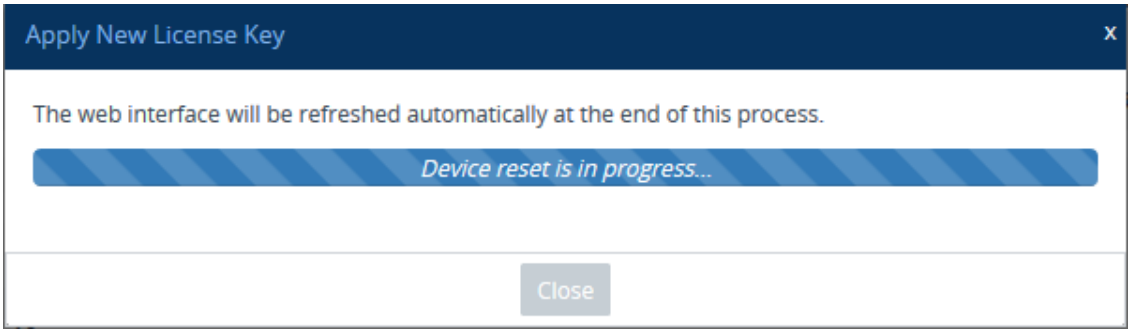


6. Click **Apply**; the dialog box closes and the "String Uploaded!" message is briefly displayed at the bottom of the page when the License Key is successfully loaded to the device. The License Key page uses color-coded icons to indicate the changes between the currently installed License Key and the newly loaded License Key. For more information, see Installing License Key through Web Interface.

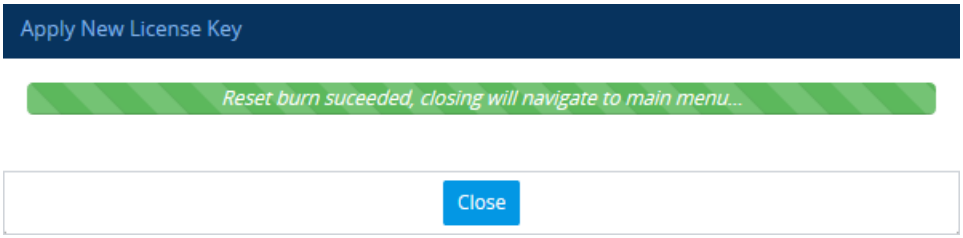
7. Click **Apply New License Key**; the following message box appears:



8. Click **Reset**; the device saves the file to flash memory with a device reset, displaying the following progress message box:



When installation completes, the following message box appears:



9. Click **Close** to close the message box; you are logged out of the Web interface and prompted to log in again. The features and capabilities displayed on the License Key page now reflect the newly installed License Key.

Installing a License Key File

You can install the License Key as a file through the Web interface. The procedure depends if the device is in standalone mode (see Installing on Standalone Devices below) or operating in HA mode (see Installing on HA Devices on the next page).

Installing on Standalone Devices

You can install the License Key as a file through the Web interface when the device operates as a standalone device. For standalone devices, you can also install the License Key as a string, as described in Installing a License Key String on page 874.

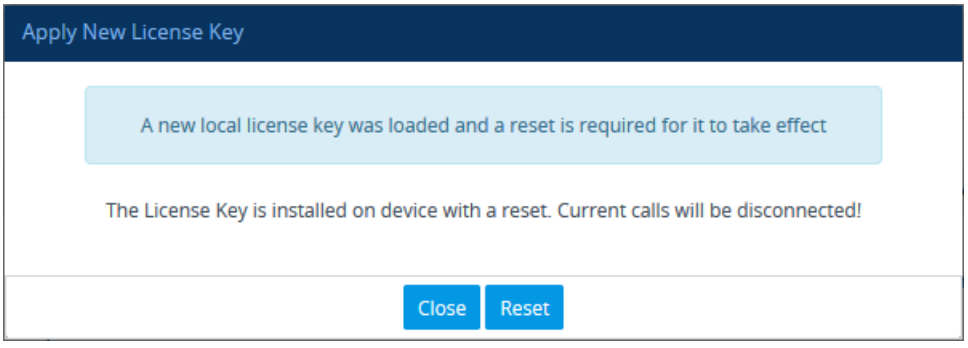
> The License Key installation process includes a device reset and is therefore, traffic-affecting. To minimize disruption of current calls, it is recommended to perform this procedure during periods of low traffic.

➢ **To install License Key file through Web interface:**

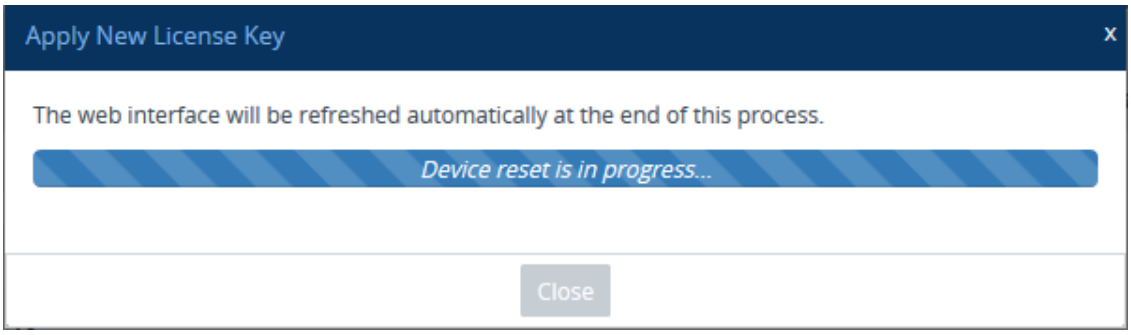
1. Place the purchased License Key file in a folder on the computer from where you are logged into the device.
2. Open the License Key page (see Viewing the License Key on page 871).
3. Back up the currently installed License Key as a precaution. If the new License Key does not comply with your requirements, you can re-load the backed-up License Key to restore the device's original capabilities. For backing up the License Key, see Backing up Local License Key on page 880.
4. Click the **Load File** button to select the License Key file on your computer; the **Apply New License Key** button appears. The License Key page uses color-coded icons to indicate the changes between the currently installed License Key and the newly loaded License Key (see Installing License Key through Web Interface on page 873).

> If want to cancel installation, reset the device without a save to flash. For more information, see Resetting the Device.

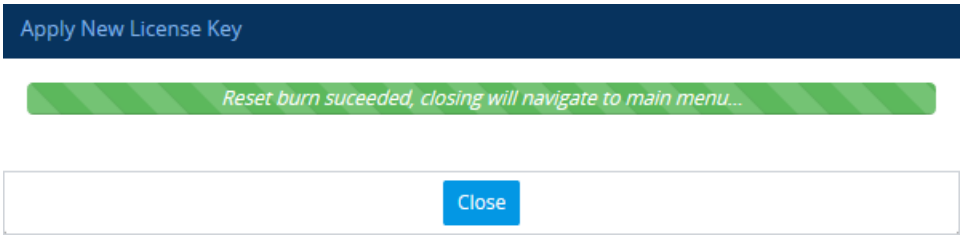
5. Click **Apply New License Key**; the following message box appears:



6. Click **Reset**; the device saves the file to flash memory with a reset and the following progress message box appears:



When installation completes, the following message box appears:



7. Clock **Close** to close the message box; you are logged out of the Web interface and prompted to log in again. The features and capabilities displayed on the License Key page now reflect the newly installed License Key.

Installing on HA Devices

When the device operates in HA mode, you can **only** install the License Key from a License Key file.

> The License Key file for HA contains two License Keys - one for the active device and one for the redundant device. Each License Key has a different serial number ("S/N"), which reflects the serial number of each device in the HA system.

You can install the License Key using one of the following methods:

- **Hitless Upgrade:** This method allows you to install the License Key on both active and redundant devices without affecting traffic. The installation process is as follows:
  a. The License Key file is loaded to the active device.
  b. The active device sends the file to the redundant device.
  c. The redundant device installs the file and saves it to flash memory with a reset.
  d. The active device saves the file to flash memory (**but without** a reset).
  e. The devices undergo an HA switchover, whereby the active device becomes redundant and the redundant device becomes active. Current calls are maintained and handled by the active device (previously the redundant device).
  f. The redundant device (previously the active device) resets to install the file.

Hitless Upgrade of the License Key is applicable only if the new License Key includes changes, compared to the currently installed License Key, only within the limited set of features below:

- FEU
- SBC
- Coder Transcoding
- SBC Signaling

- **Non-hitless Upgrade:** This method allows you to install the License Key simultaneously on both active and redundant devices (both reset at the same time). Therefore, this method is traffic-affecting and current calls are terminated. The installation process is as follows:

  a. The License Key file is loaded to the active device.

  b. The active device sends the file to the redundant device.

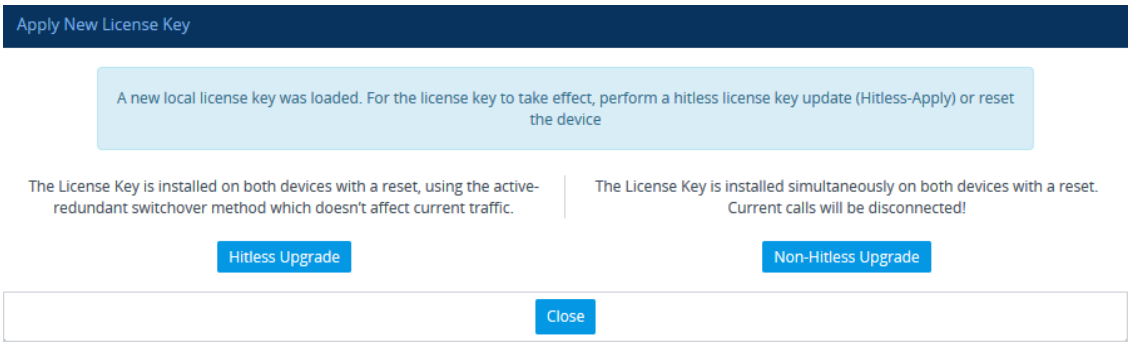
  c. Both devices install the file and save it to flash memory with a reset.

➢ To install License Key file for HA through Web interface:

1. Place the purchased License Key file in a folder on the computer from where you are logged into the device.
2. Open the License Key page (see Viewing the License Key on page 871).
3. Back up the currently installed License Key as a precaution. If the new License Key does not comply with your requirements, you can re-load this backed-up License Key to restore the device's original capabilities. For backing up the License Key, see Backing up Local License Key on page 880.
4. Click the **Load File** button to select the License Key file on your computer; the **Apply New License Key** button appears. The License Key page uses color-coded icons to indicate the changes between the currently installed License Key and the newly loaded License Key (see Installing License Key through Web Interface on page 873).

If want to cancel the License key installation, reset the device without a save to flash. For more information, see Resetting the Device.

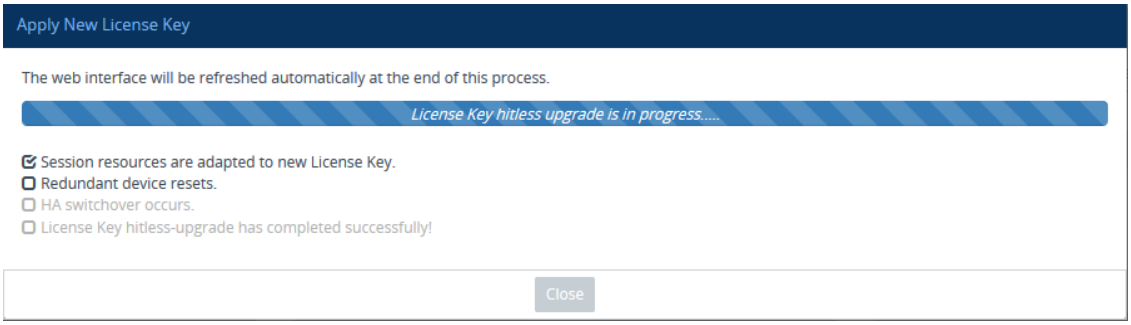
5. Click **Apply New License Key**; the following appears:



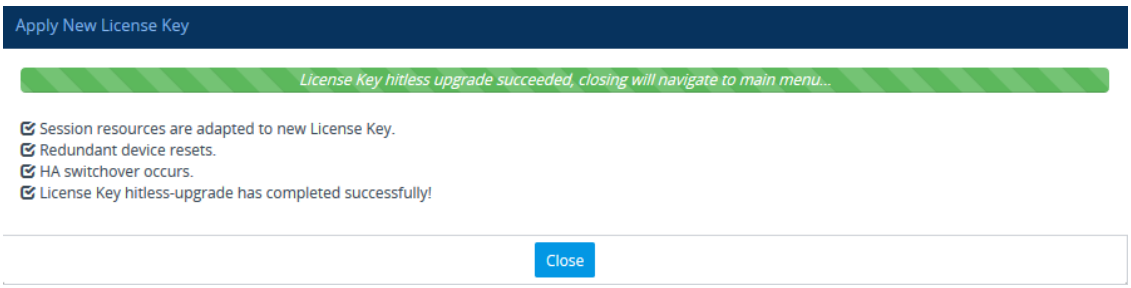
If the new License Key includes changes in licenses for features other than (or in addition to) FEU, SBC, Coder Transcoding, and/or SBC Signaling, then you can **only** use the **Non-Hitless Upgrade** method. Instead of the above page, the Web interface displays a page prompting you to click **Reset** (or **Close** to cancel the operation) to perform a non-hitless upgrade. In this scenario, skip Step 5.

6. Click one of the following buttons:

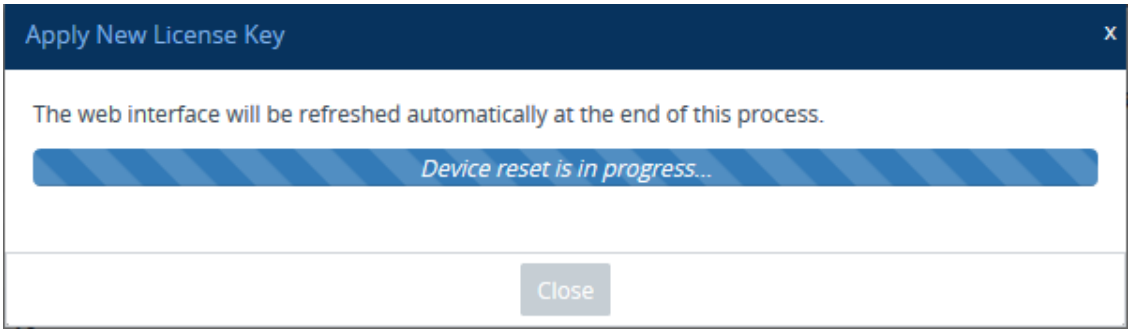
- **Hitless Upgrade:** Installs the License Key without affecting traffic by employing the HA switchover mechanism. When you click the button, the process starts and a message box is displayed indicating the installation progress:



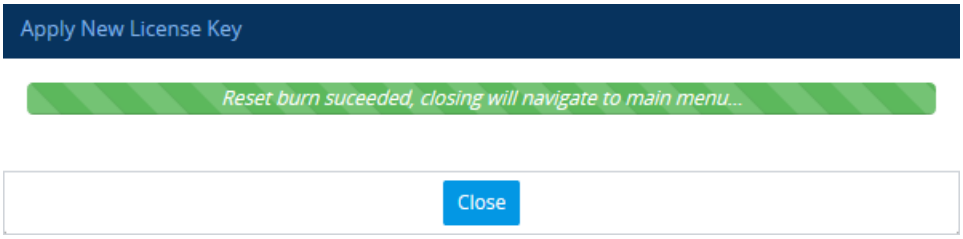
When installation completes, the following message box appears:



- **Non-Hitless Upgrade:** Installs the License Key simultaneously on both devices where both undergo a reset and therefore, current calls are terminated. When you click the button, the process starts and the following progress message box appears:



When installation completes, the following message box appears:



7. Clock **Close** to close the message box; you are logged out of the Web interface and prompted to log in again. The features and capabilities displayed on the License Key page now reflect the newly installed License Key.

Installing License Key String through CLI

To install the License Key string through CLI, use the following command:

```
(config-system)# feature-key <"License Key string enclosed in double quotation marks">
```

To view the installed License Key, use the following command:

```
show system feature-key
```

Verifying Installed License Key

To verify that the new License Key has been installed:

1. On the License Key page, check that the listed features and capabilities of the new License Key match those that you ordered.
2. Access the Syslog server and check that the following message appears:

   "S/N<serial number> Key Was Updated. The Board Needs to be Reloaded with ini file\n"

   If the Syslog server indicates that the License Key was unsuccessfully loaded (i.e., the "SN_" line is blank), do the following preliminary troubleshooting procedures:

   a. Open the License Key file and check that the "S/N" line appears. If it does not appear, contact your AudioCodes sales representative.
   b. Verify that you have loaded the correct file. Open the file and ensure that the first line displays "[LicenseKeys]".
   c. Verify that the content of the file has not been altered.

Backing up Local License Key

You can back up the installed local License Key. This may be useful, for example, if you have installed a new License Key and you want to revert to the previous License Key.

➢ **To back up local License Key:**

1. Open the License Key page (see Viewing the License Key on page 871).
2. Click one of the following buttons:
   - : Saves the License Key as a file to a folder on your computer. By default, the device saves the License Key as a .txt file type with the name license.txt.
   - : Copies the License Key as a string to your computer's clipboard. You can then paste the string into, for example, an e-mail message or a text-based program such as Notepad.

OVOC-Managed SBC Capacity Licenses

The device's licenses for SBC capacity -- SBC sessions, transcoding sessions, and user registrations -- can be provided by and managed remotely by OVOC.

OVOC offers the following SBC capacity licensing models:

- Fixed License Pool (see Fixed License Pool Model below)
- Floating License (Floating License Model on page 884)
- Flex License (Flex License Model on page 886)

Fixed License Pool Model

The device can receive SBC capacity licenses from a centralized pool of SBC licenses that is located on and managed by AudioCodes OVOC management tool. The license pool is purchased for OVOC as one bulk license and is used to provide SBC licenses to multiple devices. The OVOC user manually allocates a specific number of SBC licenses per license type (see list below) from the pool to each device in the network. Whenever required, the OVOC user can increase or decrease the number of allocated SBC licenses according to the device's capacity demands. The allocation of licenses to the devices cannot exceed the purchased Fixed License pool.

The Fixed License pool includes the following SBC capacity license types:

- SBC Sessions (maximum number of concurrent SBC call sessions - media and signaling)
- Far End Users (maximum number of SIP endpoints or users that can be registered with the device)
- Transcoding Sessions (maximum number of transcoding sessions)

- The Fixed License does not involve any configuration on the device; it is enabled and managed entirely by OVOC. For more information on the OVOC License Pool, refer to the *OVOC User's Manual*.

As an example of how the Fixed License pool allocates licenses, assume that the pool contains a maximum of 20 SBC far-end user (registration) licenses and it needs to service three devices (A, B and C). It can allocate 10 to A, 8 to B, and 2 to C. In this example, because all the far-end user licenses in the pool have been allocated, it cannot allocate any more far-end user licenses to the devices. However, if it de-allocates 5 licenses from A, for example, it can allocate these additional licenses to B and/or C.

As another example, assume that an OVOC tenant is allocated 500 SBC Session licenses to service 4 devices (A, B, and C), where each device has a capacity of 250 SBC sessions. If A and B are operating at maximum capacity (i.e. aggregated number of active SBC call sessions is 500), and C requires 50 SBC sessions, then C is taken out-of-service until the number of active calls on A and B combined drops to 450 sessions. When this occurs, the 50 free licenses can be allocated by the pool to C. If over a period of time, call traffic on A and B is showing a downward trend, the OVOC user can reallocate extra licenses to C.

The device periodically (and after a device reset or HA switchover if operating in HA mode) checks with OVOC for any SBC capacity license updates. OVOC identifies the device by serial number and sends licenses to the device according to OVOC configuration. If the device's local License Key already includes SBC capacity licenses, the SBC licenses allocated by OVOC are added to it (but up to the device's maximum supported capacity capabilities). When the device applies the licenses received from OVOC, the License Key page displays "License Pool" in the 'Mode' field (see Viewing the License Key on page 871) and displays the allocated SBC licenses under the **SBC Capacity** group, as shown in the example below:

SBC CAPACITY

	Remote	Local	Actual
SBC Sessions	5	10	15
SBC Signaling Sessions	2	5	7
SBC Media Sessions		5	5
Far End Users (FEU)	2	22	24
Transcoding Sessions	2	20	22

- 'Remote': This column displays the number of SBC licenses per license type received from the OVOC Fixed License pool.
- 'Local': This column displays the number of SBC licenses per license type from the locally installed License Key.
- 'Actual': This column displays the total SBC licenses per license type, which is the summation of the remote and local licenses.

(Standalone devices only) For the SBC licenses allocated by OVOC to take effect, the device **must** be reset with a save to flash. The reset can be initiated by the OVOC user or locally on the device by you. The total licenses (displayed in the "Actual" column) is only updated once the device completes this reset.

Communication between the device and OVOC is through HTTPS (port 443) and SNMP. If a firewall exists in the network, make sure that ports for these applications are opened. If the device loses connectivity with OVOC for a long duration, it discards the allocated SBC licenses and resets with its initial SBC licenses according to the local License Key. This mechanism prevents misuse of SBC licenses allocated by the OVOC license pool. Connectivity status with OVOC is displayed in the 'License Server Status' field on the License Key page (see Viewing the License Key on page 871).

When the device operates in High-Availability (HA) mode, the SBC licenses allocated from the OVOC Fixed License pool are installed on the active and redundant devices without affecting traffic. This is achieved by the Hitless Upgrade mechanism:

1. OVOC downloads the licenses to the redundant device, and then resets it.
2. OVOC triggers an HA switchover.
3. OVOC downloads the same licenses to the previously active device (now redundant), and then resets it.
4. OVOC triggers another HA switchover.

The device sends the following SNMP alarms to indicate various conditions relating to the allocation of SBC licenses by the OVOC Fixed License pool:

- acLicensePoolInfraAlarm (OID 1.3.6.1.4.1.5003.9.10.1.21.2.0.106): Sent if the device loses connection with OVOC, for example.
- acLicensePoolApplicationAlarm (OID 1.3.6.1.4.1.5003.9.10.1.21.2.0.107): Sent when the device receives new SBC licenses from the Fixed License pool.
- acLicensePoolOverAllocationAlarm (OID 1.3.6.1.4.1.5003.9.10.1.21.2.0.125): Sent when the device receives new SBC licenses from the Fixed License pool, which has caused the device to exceed its maximum supported capacity.

For more information on these alarms, refer to the device's *SNMP Reference Guide*.

- The Fixed License only provides SBC capacity licenses (listed in the beginning of this section). Therefore, your device must still be installed with a local License Key to enable other ordered license-based features (e.g., Test Calls) and capacity.
- The allocation and de-allocation of SBC licenses to standalone devices by the OVOC Fixed License pool is service affecting because it requires a device reset.
- For HA systems, the OVOC Fixed License pool automatically allocates an equal number of SBC licenses to the active and redundant devices. For example, if the license pool allocates 200 signaling sessions to the active device, it also allocates 200 to the redundant. Therefore, it is important to take this into consideration when ordering a license pool.
- If the device is restored to factory defaults, the SBC licenses allocated by the OVOC Fixed License pool are removed and only the SBC licenses from the locally installed License Key are applied instead.
- If the device is allocated SBC licenses by the OVOC Fixed Licenses pool that exceeds the maximum number of sessions that the device can support, the device sets the number of sessions to its maximum supported capacity.
- The Fixed License pool cannot operate with the other OVOC-managed license modes (e.g., Floating License). Therefore, before using the Fixed License, make sure that the other license modes are disabled on the device and OVOC.

Floating License Model

The Floating License is a network-wide SBC capacity-related license pool, which is managed by AudioCodes OVOC and the cloud-based License Manager, and shared dynamically among multiple devices. The Floating License is a pay-as-you-grow service, eliminating the need to manually purchase additional SBC licenses each time your capacity requirements increase. You initially purchase a Floating License based on your estimated SBC capacity requirements. If you later experience business growth and your devices use more SBC licenses than specified by the Floating License, you are billed for these additional licenses. In other words, the Floating License pool capacity can be exceeded.

The Floating License pool includes the following SBC capacity license types:

- SBC sessions (maximum number of concurrent SBC call sessions - media and signaling)
- User registrations (maximum number of SIP endpoints that can be registered with the device)
- Transcoding sessions (maximum number of transcoding sessions)

As an example of how the Floating License pool operates, assume that an OVOC tenant is allocated 500 SBC Session licenses and the tenant has deployed three devices (A, B, and C), where each device has a maximum capacity of 250 SBC sessions. If A and B are operating at maximum capacity (i.e. the aggregated number of active SBC call sessions is 500), and then C requires 50 SBC sessions, even though the initially purchased Floating License pool capacity has been reached (500), C is allowed to process these 50 new call sessions. When you are next billed, you are charged for these extra 50 SBC session licenses.

For providing the Floating License service, OVOC and the Cloud License Manager need to be set up accordingly (refer to the *OVOC User's Manual*). The Floating License service also needs to be enabled on these devices. Once these devices connect to OVOC, they are "open" to use any number of licenses in the Floating License pool. However, capacity is limited by the device's inherent maximum capacity support and by an optional user-defined limit called *Allocation Profile* (discussed later in this section), which specifies a capacity that is less than the device's inherent capacity per SBC license type.

Connection between the devices and OVOC is established over SNMP. Functionality of the Floating License service is managed over TCP/HTTPS REST. For more information, see the *One Voice Operations Center IOM Manual* and the *OVOC Security Guidelines*. Connectivity status with OVOC is displayed in the 'License Server Status' field on the License Key page (see Viewing the License Key on page 871). If the device loses connectivity with OVOC, it continues using the licenses that it received before the disconnection for a specific grace period, and then once this period expires, it stops accepting new calls.

The devices report their SBC license consumption per license type to OVOC at fixed intervals (typically, every five minutes). OVOC accumulates these reports and sends them to AudioCodes Cloud License Manager every 12 hours with all the SBC licenses usage in the last 12 hours. OVOC uses REST APIs over HTTPS to report to the Cloud License Manager. AudioCodes personnel analyze these license consumption reports in the Cloud License Manager on a monthly basis to check if capacity specified by your Floating License was exceeded. If it was exceeded, AudioCodes sends you a report detailing the excess licenses and requests that you purchase additional SBC licenses for your Floating License. To view the Floating License reports of SBC license consumption that the device sends OVOC, see Viewing Floating or Flex License Reports on page 893.

When the device uses the Floating License, the License Key page (see Viewing the License Key on page 871) displays "Floating License" in the 'Mode' field and display the SBC capacity licenses received from the Floating License under the **SBC Capacity** group, as shown in the example below:

SBC CAPACITY

	Local	Floating	Actual
Far End Users	~~5~~	0	0
SBC Media Sessions		250	250
SBC Signaling Sessions		250	250
Transcoding Sessions		0	0
SBC Sessions	~~2~~		

- 'Local': This column displays the number of SBC licenses per license type from the locally installed License Key. These licenses are not used by the device and the figures are displayed crossed out (strikethrough).

- 'Floating': This column displays the number of SBC licenses per license type received from the OVOC Floating License pool.
- 'Actual': (see the 'Floating' column).

The device sends the following SNMP alarms to indicate various conditions relating to the allocation of SBC licenses by the OVOC Floating License pool:

- acFloatingLicenseAlarm: Sent if you have configured an Allocation Profile that exceeds the device's maximum supported capacity.
- acCloudLicenseManagerAlarm: Sent upon various conditions such as loss of connectivity between the device and OVOC.

For more information on these alarms, refer to the device's *SNMP Reference Guide*.

- The Floating License only provides SBC capacity licenses (listed previously). Therefore, your device must still be installed with a local License Key to enable the other ordered license-based features (e.g., Test Calls) and capacity.
- For configuring the Floating License on OVOC, refer to the *OVOC User's Manual*.
- The Floating License cannot operate with other OVOC-managed SBC capacity license modes (e.g., Fixed License). Therefore, before enabling the Floating License, make sure that the other license modes are disabled on OVOC.
- The Floating License ignores OVR,, and LAD capacity licenses in the local License Key.

Flex License Model

The Flex License model is a network-wide SBC capacity-related license, managed by AudioCodes OVOC, which is dynamically shared among multiple devices. The Flex License is ordered as a single license, which provides a pool of SBC licenses that cannot be exceeded. The Flex License pool includes the following SBC capacity license types:

- SBC Sessions (maximum number of concurrent SBC call sessions - media and signaling)
- Far End Users (maximum number of SIP endpoints or users that can be registered with the device)
- Transcoding Sessions (maximum number of transcoding sessions)

The Flex License model is similar to the Floating License model (as described in Floating License Model on page 884), but provides some important advantages:

- The Flex License is solely managed by OVOC; it doesn't employ a cloud-based license manager like the Floating License. This reduces the exposure of OVOC to security risks from its connectivity with the public cloud.
- The Flex License gracefully enforces license capacity of the pool; the Floating License allows devices to exceed pool capacity, resulting in you being billed at the end of the month for unexpected license usages.

The Flex License is managed by AudioCodes OVOC, which defines the devices using the Flex License. Once connected to OVOC, each device can handle calls using the licenses of the different license types in the Flex License pool, as long as the pool has available (unused) licenses. However, the device's capacity is limited by its inherent maximum capacity support and by an optional user-defined limit called *Allocation Profile* (discussed later in this section), which specifies a capacity that is less than the device's inherent capacity per SBC license type.

The devices periodically (typically, every five minutes) report their current SBC license consumption (usage) per SBC license type to OVOC. OVOC uses these reports to calculate the total number of currently used licenses from the pool and therefore, determines the remaining licenses in the pool per license type. To view the license usage reports that the device sends to OVOC, see Viewing Floating or Flex License Reports on page 893.

Each device in OVOC is configured with a priority level (Low, Normal, or Critical). When all the licenses of a specific license type in the Flex License pool are being used (or even exceeded) by the devices, OVOC uses this priority level to determine which of the devices to initially "take out" of service. OVOC first notifies a certain percentage of devices of this "over-license" status, instructing them to **reject all new calls** that require this specific license type. This percentage of devices starts from those with Low priority level, then Normal priority level, and lastly Critical priority level.

For example, assume there are 100 devices in the network, 10 configured with Low priority, 20 with Normal priority, and 70 with Critical priority, and OVOC notifies 20% of them of an "over-license" state for a specific license type. In this example, OVOC takes out of service the 10 devices with Low priority and 10 devices with Normal priority (i.e., total of 20, which is 20% of 100). This selective process allows devices with higher priority to continue providing call service, while attempting to restore licenses to the Flex License pool due to the rejection of new calls by the selected devices. During this period, the devices send their usage reports more frequently to OVOC, providing OVOC with a more up-to-date status of license usages in the network. If licenses become available for the specific license type in the pool, OVOC allows the selected devices to start accepting new calls ("ok" status). However, if after a certain period there are still unavailable licenses for the specific license type in the pool, OVOC notifies all devices (including those with Critical priority level) of this "over-license" status, and instructs **all of them to reject new calls**. To view the device's current license utilization (in percentage) per license type of the OVOC Flex License pool and the status ("ok" and "overlicense") of each license type, see Viewing Flex License Utilization and Status on page 890.

Connection between the devices and OVOC is established over SNMP and functionality of the Flex License service is managed over TCP/HTTPS REST. If the device loses connectivity with OVOC, the device continues handling calls for a graceful period. If connectivity is not restored when this period expires, the device is blocked from handling new calls. When the device succeeds in connecting again with OVOC, it continues using the Flex License pool as normal.

When the device uses the Flex License, the License Key page (see Viewing the License Key on page 871) displays "Flex License" in the 'Mode' field and displays the SBC capacity licenses received from the Flex License pool per license type under the **SBC Capacity** group, as shown in the following example:

SBC CAPACITY

	Local	Flex	Actual
Far End Users	~~240~~	10	200
SBC Media Sessions		9	20
SBC Signaling Sessions		12	60
Transcoding Sessions		5	120
SBC Sessions	~~240~~		

- 'Local': This column displays the number of SBC licenses per license type from the locally installed License Key. These licenses are not used by the device and the figures are displayed crossed out (strikethrough).
- 'Flex': This column displays the maximum number of SBC licenses per license type in the OVOC Flex License pool.
- 'Actual': (This column can be ignored.)

> After a device reset or HA switchover, the figures in the 'Flex' column appear only after the device receives its first report from OVOC on the Flex License pool capacity. This typically takes about five minutes.

The device sends the following SNMP alarms to indicate various conditions relating to the OVOC Flex License pool:

- **acFloatingLicenseAlarm:** Sent if you have configured an Allocation Profile that exceeds the device's resource capability.
- **acCloudLicenseManagerAlarm:** Sent upon various conditions such as loss of connectivity between device and OVOC.

For more information on these alarms, refer to the device's *SNMP Reference Guide*.

> - For configuring the Flex License on OVOC, refer to the *OVOC User's Manual*, which can be downloaded from AudioCodes website.
> - The Fixed License only provides SBC capacity licenses (listed in the beginning of this section). Therefore, your device must still be installed with a local License Key to enable other ordered license-based features (e.g., Test Call) and capacity.
> - The Flex License cannot operate with the other OVOC-managed license modes (e.g., Fixed License and Floating License). Therefore, before enabling the Flex License, make sure that the other license modes are disabled on OVOC.
> - The Flex License ignores OVR, and LAD capacity licenses in the local License Key.

Enabling Floating or Flex License

Before you can use the Floating or Flex license, you need to enable this feature.

Prior to enabling the Floating or Flex License, make sure that the following OVOC-related prerequisites have been fulfilled:

- The Floating or Flex License has been purchased from AudioCodes with the required SBC license capacities and installed on OVOC.
- The devices for which you want to use the Floating or Flex License have been configured on OVOC.
- For Floating License, OVOC has been configured for communication with AudioCodes Cloud License Manager.
- For Flex License, the devices have been configured with priority levels on OVOC.

For more information on configuring and managing the Floating or Flex License on OVOC, refer to the *OVOC User's Manual*, which can be downloaded from AudioCodes website.

➢ **To enable the Floating or Flex License:**

1. Open the Floating License page (**Setup** menu > **Administration** tab > **License** folder > **Floating License**).
2. From the 'Floating License' drop-down list, select **Enable**.

Figure 36-1: Enabling the Floating or Flex License

3. Reset the device with a burn-to-flash for your settings to take effect. After the device resets, it connects with OVOC and OVOC-related information is displayed in the read-only fields:
   - **'Connection with OVOC':** Displays the device's connectivity status with OVOC:
     - "Connected": The device is connected to OVOC.
     - "Disconnected" The device has disconnected from OVOC due to problems with the network (HTTPS TCP connection).
     - "Not Connected": The device is not connected to OVOC.
   - **'OVOC IP Address':** Displays the IP address of OVOC.

- **'OVOC Product Key':** Displays the Product Key of OVOC that is providing the Floating or Flex License.

> Once you enable the Floating or Flex License, OVOC initiates a connection with the device. In other words, you don't configure the address of OVOC. The device connects with OVOC over SNMP and an SNMP manager is automatically added to the SNMP Trap Destinations table for this connection (see Configuring SNMP Trap Destinations with IP Addresses on page 80).

The status of the Floating License or Flex License is also displayed on the top bar of the License Key page, as shown below (e.g., Flex License mode):

- **'OVOC Product Key':** Product Key of the OVOC tool providing the Floating License or Flex License
- **'Mode':** Indicates the license type:
  - "Floating License": Floating License mode
  - "Flex License": Flex License mode
- **'License Server Status':** Connectivity status with OVOC (for more information, see Viewing the License Key on page 871)

Viewing Flex License Utilization and Status

You can view the device's current license utilization (in percentage) per license type of the OVOC Flex License pool.

You can also view the status of total license utilization per license type of the pool by **all** devices. If total utilization is within the pool's capacity, the "ok" status is displayed. If utilization has reached (or exceeded) the pool's capacity, the "overlicense" status is displayed. When the status is "overlicense", OVOC first attempts to return licenses to the pool by instructing certain devices (based on their priority level) to block new calls. If this doesn't help after a graceful period, OVOC instructs all devices to block new calls until licenses return to the pool. For more information, see Flex License Model on page 886.

> - The **Flex Pool** group (below) appears only if you have enabled the Floating License or Flex License feature (see Floating License Model on page 884).

➢ **To view Flex License utilization and status:**

1. Open the Floating License page (**Setup** menu > **Administration** tab > **License** folder > **Floating License**).
2. Scroll down to the **Flex Pool** group:

FLEX POOL

	Utilization	Status
SBC Media Sessions	66.67%	ok
SBC Signaling Sessions	50.00%	ok
Far End Users	<1%	ok
Transcoding Sessions	120.00%	overlicense

- 'Utilization': Displays the percentage (%) of the license capacity per license type in the OVOC Flex License pool that the device is currently using. Utilization of less than 1% is displayed as "<1%".
- 'Status': Displays the utilization status of the OVOC Flex License pool by all devices:
  - "ok": Utilization is within the pool capacity.
  - "overlicense": Utilization has reached or exceeded the Flex License pool capacity.

Configuring Floating or Flex License Allocation Profiles

The Floating or Flex License allows you to configure *Allocation Profiles*, which specify license capacity per license type that you want allocated to the device by OVOC. For example, you may want to limit the device to only 20 Far End Users, even though OVOC could allocate up to 100 Far End Users.

You can choose a default Allocation Profile (**SIP Trunking** or **Registered Users**) that has a pre-defined capacity suited for these applications, or you can configure a customized Allocation Profile. In addition, once you have chosen an Application Profile and reset the device to apply it, you can easily reduce (*limit*) the pre-defined capacity or customized capacity when needed, without resetting the device.

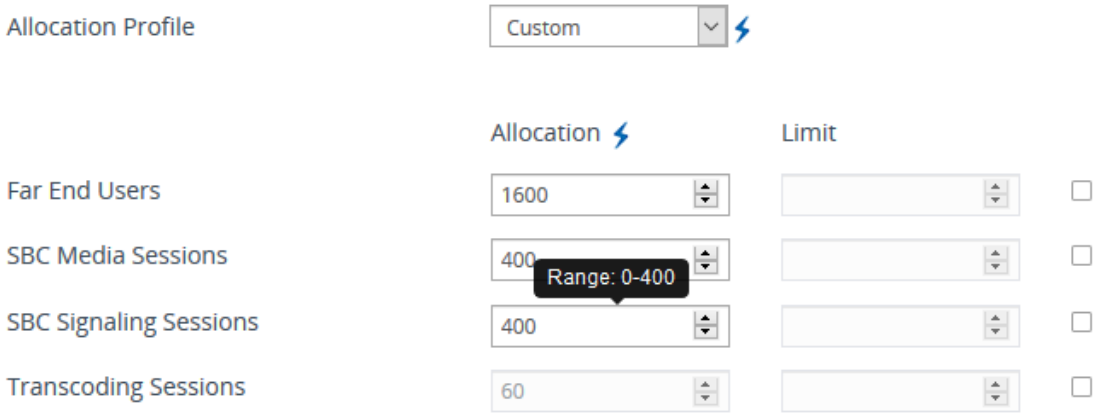
You can only configure Allocation Profiles once you have enabled the Floating or Flex License (see Enabling Floating or Flex License on page 889).

➢ **To configure Allocation Profiles:**

1. Open the Floating License page (**Setup** menu > **Administration** tab > **License** folder > **Floating License**).
2. From the 'Allocation Profile' drop-down list, select an SBC license Allocation Profile:
   - **SIP Trunking:** Provides default capacity (cannot be modified) in the 'Allocation' field per license type and is suited for SIP Trunking applications (i.e., where user registration is typically not required). You can later reduce the capacity using the 'Limit' field after you reset the device, as described in Step 4.

- **Registered Users:** Provides default capacity (cannot be modified) in the 'Allocation' field per license type and is suited for applications where user registration is required. You can later reduce the capacity using the 'Limit' field after you reset the device, as described in Step 4.
- **Custom:** Allows you to configure a customized Allocation Profile. In the 'Allocation' field corresponding to each SBC license type, configure the desired capacity.

Figure 36-2: Configuring Allocation Profile (e.g., Custom)



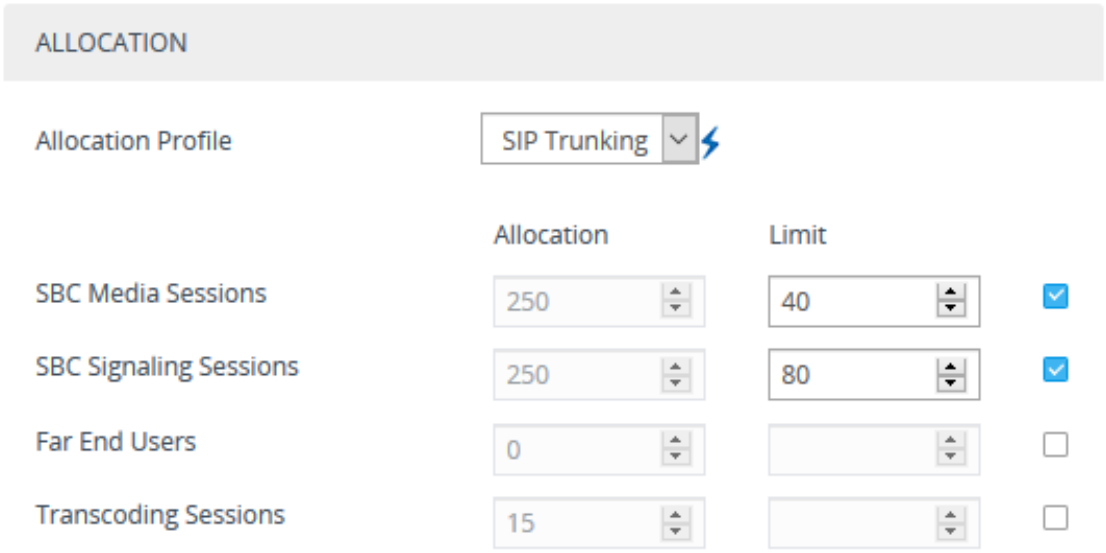
When configuring a customized Allocation Profile (i.e., 'Allocation Profile' configured to **Custom**):

- To view the device's maximum supported capacity per license type, hover your mouse over the corresponding 'Allocation' field and a pop-up appears displaying the capacity.
- The 'Transcoding Sessions' license type capacity cannot be modified in the 'Allocation' field. However, you can reduce the license using its corresponding 'Limit' field, as described below.

3. Reset the device with a burn-to-flash for your settings to take effect.
4. You can now reduce each SBC license type capacity whenever needed **without** resetting the device:
   a. Select the check box corresponding to the license type you want reduced.
   b. In the corresponding 'Limit' field, enter a new capacity. The value can be equal to or less than the value in the 'Allocation' field.
   c. Click **Apply**.

   The figure below shows an example of using the 'Limit' field to reduce the allocation of SBC Media Sessions to 40 and the SBC Signaling Sessions to 80 for the **SIP Trunking** Allocation Profile:

Figure 36-3: Configuring Limits for Allocation Profile



Viewing Floating or Flex License Reports

You can view the SBC resource consumption (signaling sessions, media sessions, transcoding sessions, and far-end user registrations) reports of the Floating License or Flex License that the device periodically sends to OVOC.

- The Floating License Reports page is available only if you have enabled the Floating License or Flex License feature (see Floating License Model on page 884).

➢ **To view the Floating License or Flex License Report through Web interface:**

- Open the Floating License Reports page (**Setup** menu > **Administration** tab > **License** folder > **Floating License Reports**.

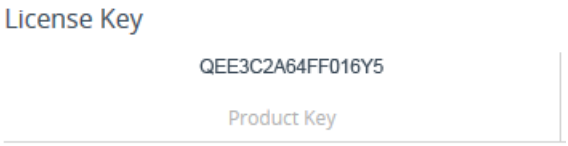
REPORT DATE	SIGNALING SESSIONS	MEDIA SESSIONS	TRANSCODING SESSIONS	FAR END USERS
2018-09-04 15:16:12	0	0	0	0
2018-09-04 15:15:11	0	0	0	0
2018-09-04 15:14:10	0	0	0	0
2018-09-04 15:13:10	0	0	0	0
2018-09-04 15:12:08	0	0	0	0
2018-09-04 15:11:08	0	0	0	0
2018-09-04 15:10:07	0	0	0	0
2018-09-04 15:09:06	0	0	0	0
2018-09-04 15:08:06	0	0	0	0
2018-09-04 15:07:04	0	0	0	0
2018-09-04 15:06:04	2111	2109	0	0
2018-09-04 15:05:03	2032	2029	0	0
2018-09-04 15:04:02	2009	2008	0	0
2018-09-04 15:03:01	2012	2010	0	0
2018-09-04 15:02:00	2009	2007	0	0

Viewing the Device's Product Key

The Product Key identifies a specific purchase of your device installation for the purpose of subsequent communication with AudioCodes (e.g., for support and software upgrades). The Product Key is your chassis' serial number--"S/N(Product Key)"--which also appears on the product label affixed to the chassis.

The Product Key is included in the License Key. You can view the Product Key on the following Web pages:

- License Key page (see Viewing the License Key on page 871). The Product Key is displayed in the read-only 'Product Key' field, as shown in the example below:



- Device Information page (see Viewing Device Information on page 945).

If your License Key was purchased in an earlier version, the 'Product Key' field may appear empty.

37 Configuration File

This section describes how to save the device's configuration to a file and how to load a configuration file to the device.

Downloading and Loading ini Configuration File

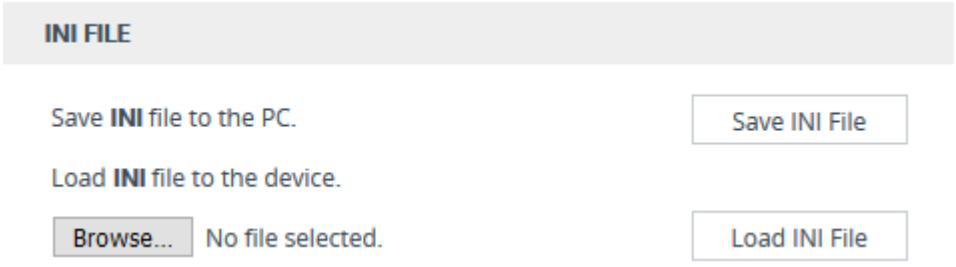
You can save (download) the device's configuration as an ini file to a folder on your computer or load (upload) an ini file to the device. Saving an ini file can serve as a backup of your configuration and if needed, you can later load the file to the device to restore your previous configuration settings.

- The saved (downloaded) ini file includes only the following:
  - ✓ Configuration tables that contain row entries (default and non-default).
  - ✓ Standalone parameters whose values you changed from default. However, it also includes parameters whose values you changed from non-default back to default without subsequently resetting the device. If you changed from non-default back to default but subsequently reset the device, then they'll not be included.
  - ✓ All SNMP performance monitoring MIBs whose threshold values (low or high) you changed from default. (To apply these same threshold values to other devices, load the ini file to the devices.)
  - ✓ The device's License Key. If the device is in High-Availability mode, the ini file of the Active device also includes the Redundant device's License Key.
- When loading an ini file, parameters not included in the file are **restored to default settings**. If you want to keep the device's current configuration settings and also apply the settings specified in the ini file, load the file through the Auxiliary Files page (see Loading Auxiliary Files through Web Interface).
- When loading an ini file, the device needs to reset for the parameter settings to take effect.

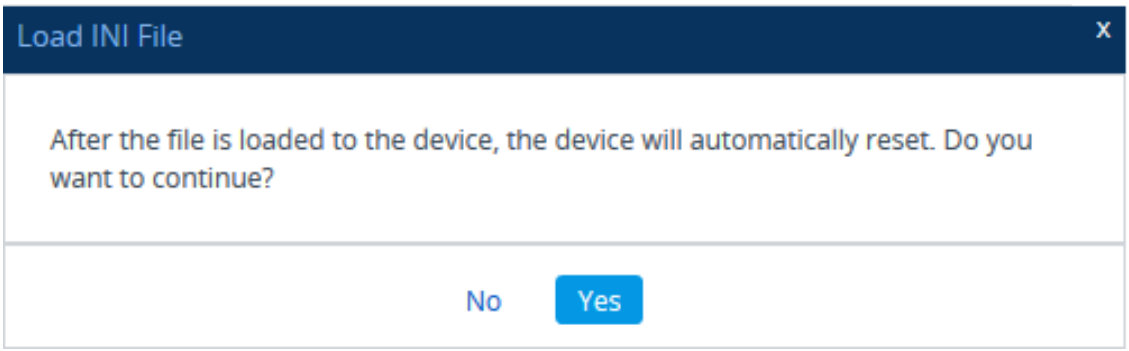
➢ **To save or load an ini file through the Web interface:**

1. Open the Configuration File page:
   - **Toolbar:** From the **Actions** drop-down menu, choose **Configuration File**
   - **Navigation tree:Setup** menu > **Administration** tab > **Maintenance** folder > **Configuration File**

   The relevant buttons for saving and loading an ini file are located under the INI File group:



2. To save the ini file: Click the **Save INI File** button, and then save the file to a folder on your computer.
3. To load an ini file:
   a. Click the **Browse** button, and then browse to and select the file on your computer.
   b. Click the **Load INI File** button; the following message box appears, informing you that the device will reset after the file is loaded.



   c. Click **Yes** to continue (or **No** to cancel the file load). If you click **Yes**, the device loads the file and then resets with a save to flash for the settings to take effect.

Saving and Loading CLI Script Files

You can save and load the device's configuration as a CLI Script file. Saving a CLI Script file can serve as a backup of your configuration and if needed, you can later load the file to the device to restore your previous configuration settings. You can also load a CLI Startup Script file.

- The CLI Startup Script file is not supported when the device operates in HA mode.
- When loading a CLI Script file, the device resets only if it contains the **reload now** command (on the last line). For more information on this command, refer to the *CLI Reference Guide*.
- When loading a CLI Startup Script file, the device needs to reset twice for its settings to take effect.
- The saved file includes only parameters whose values you have modified.
- To save the CLI Script file to a remote server (TFTP or HTTP/S): # write-and-backup to <URL with file name>

➢ To save or load a CLI Script file through Web interface:

1. Open the Configuration File page:
   - **Toolbar:** From the **Actions** drop-down menu, choose **Configuration File**
   - **Navigation tree:Setup** menu > **Administration** tab > **Maintenance** folder > **Configuration File**

2. To save the CLI Script file, under the CLI Script group: Click the **Save CLI Script File** button, and then save the file to a folder on your computer.

3. To load a CLI Script file, under the CLI Script group, do the following:

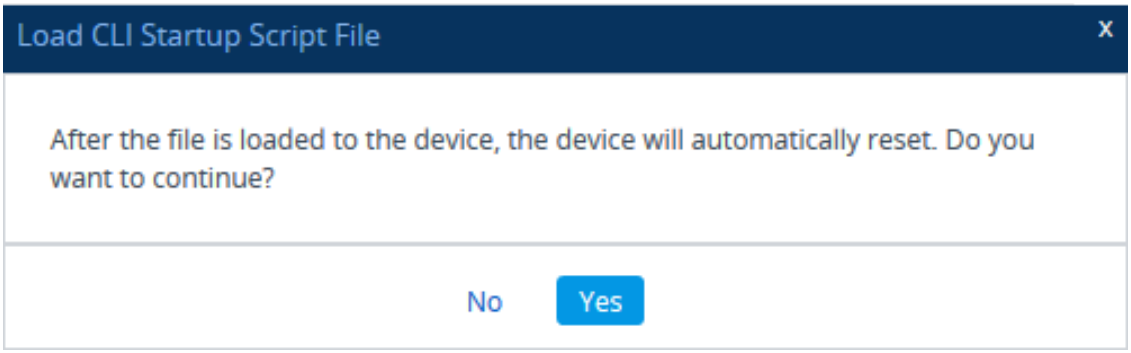
   a. Click the **Browse** button, and then browse to and select the file on your computer.

   b. Click the **Load CLI Script File** button; the device loads the file and then resets with a save to flash.

4. To load a CLI Startup Script file, under the CLI Script group, do the following:

   a. Click the **Browse** button, and then browse to and select the file on your computer.

   b. Click the **Load CLI Startup Script** button; the following message box appears, informing you that the device will reset after the file is loaded.



   c. Click **Yes** to continue (or **No** to cancel the file load). If you click **Yes**, the device loads the file and then resets with a save to flash for the settings to take effect..

38 Saving and Loading a Configuration Package File

You can save and load a bundle of files used by the device in a single, packaged file called a Configuration Package file. The packaged file is a TAR (Tape ARchive) file (.tar), compressed with gzip.

The feature can be used for backing up full configuration and then later restoring it to the device in case of device configuration failure (for whatever reason), or for loading the backed-up configuration package file to other devices requiring similar configuration.

The configuration package file can include the following files:

File	Description
ini.ini	INI configuration file.
cli-startup-script.txt	CLI Startup Script file. **Note:** This file is applicable only for uploading a Configuration Package file.
<TLS Context ID>.pkey	Private key of the TLS Context (by ID). **Note:** You can only choose to include certificates in the Configuration Package file if you enable the password-protect (encrypt) option.
<TLS Context ID>.crt	TLS certificate of the TLS Context (by ID). **Note:** You can only choose to include certificates in the Configuration Package file if you enable the password-protect (encrypt) option.
<TLS Context ID>.root	Trusted root certificate of the TLS Context (by ID). **Note:** You can only choose to include certificates in the Configuration Package file if you enable the password-protect (encrypt) option.
LOGO.dat	Image file used as the logo in the Web interface.
FAVICON.dat	Favicon file used by Web browsers to represent the device's Web interface.
CPT.dat	Call Progress Tone file (CPT).
PRT.dat	Pre-recorded Tone file (PRT).
AMD.dat	Answer Machine Detection file (AMD).
SBC_WIZARD.dat	SBC Configuration Wizard template file
DPLN.dat	Dial Plan file. **Note:** Only for backward compatibility for versions that supported a

File	Description
	Dial Plan file. For current versions, the Dial Plan is included in the ini file.
DialPlanRule.csv	Dial Plan file.

- When loading a Configuration Package file, the device needs to reset for the settings to take effect.
- You can manually add TLS certificate files (i.e., <ctx_id>.crt, <ctx_id>.root, or <ctx_id>.pkey) to an already downloaded Configuration Package file and then upload it to the device.
- For the certificate files, only the root certificate file (.root) can be saved.
- When loading a Configuration Package file, the filenames must be as listed above.
- By default, the Configuration Package file is saved with the filename "ConfBackupPkg<Serial Number>.tar.gz".
- The Configuration Package file is included in the device's debug file and core dump file (see Viewing Debug (and Core Dump) File Contents on page 1093.

You can save and load a Configuration Package file using the following methods:

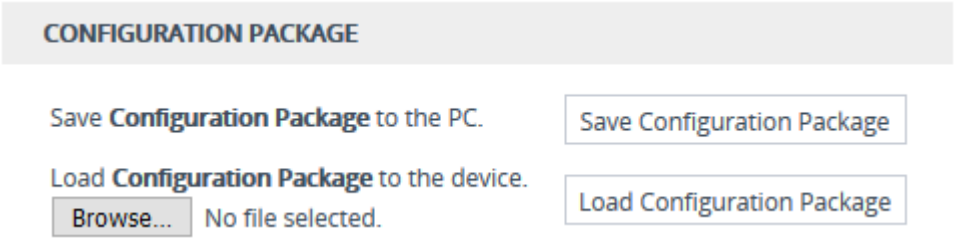
- **CLI:**

```
# copy configuration-pkg from|to <URL>
```

- **Auto-Update Feature:** To load the Configuration Package file through the Auto-Update mechanism, use the [ConfPackageURL] ini file parameter.
- **SFTP:** The Configuration Package file can also be downloaded (Get) from the device through SFTP. The file is located in the */configuration* directory. Your SFTP client needs to authenticate itself with the SFTP server (i.e., the device) and access is granted only to users with Security Administrator level.
- **Web interface:**

  a. Open the Configuration File page:

  - **Toolbar:** From the **Actions** drop-down menu, choose **Configuration File**
  - **Navigation tree:Setup** menu > **Administration** tab > **Maintenance** folder > **Configuration File**



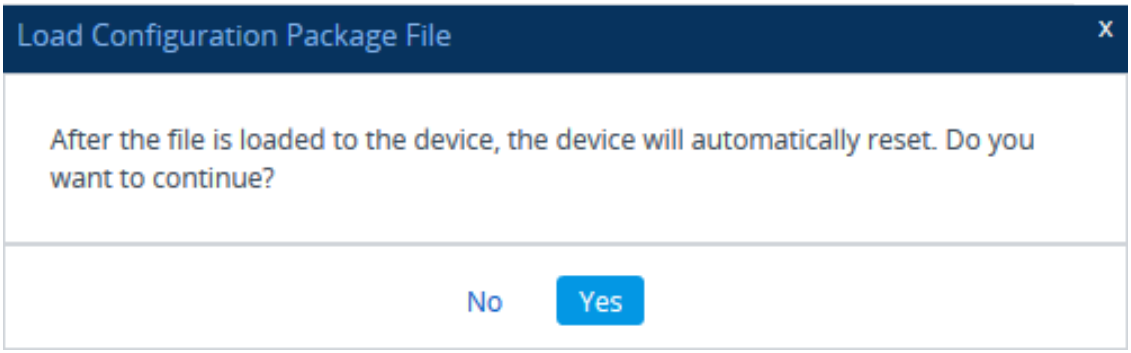
b. To download the Configuration Package file:

i. Click the **Save Configuration Package** button, and then save the file to a folder on your computer.

c. To upload a Configuration Package file:

i. Click the **Browse** button, and then browse to and select the file on your computer.

ii. Click the **Load Configuration Package** button; the following message box appears, informing you that the device will reset after the file is loaded.



iii. Click **Yes** to continue (or **No** to cancel the file load). If you click **Yes**, the device loads the file and then resets with a save to flash for the settings to take effect.

39 Automatic Provisioning

This chapter describes automatic provisioning of the device.

Automatic Configuration Methods

The device supports the following automatic provisioning methods:

- DHCP (Option 160)
- HTTP/S
- TFTP
- FTP
- SNMP (AudioCodes OVOC)

DHCP-based Provisioning

A third-party DHCP server can be configured to automatically provide each device, acting as a DHCP client, with a temporary IP address so that individual MAC addresses are not required. The DHCP server can provide additional networking parameters such as subnet mask, default gateway, primary and secondary DNS server, and two SIP server addresses. These network parameters have a time limit, after which the device must 'renew' its lease from the DHCP server.

The device can use a host name in the DHCP request. The host name is set to "acl_nnnnn", where *nnnnn* denotes the device's serial number. The serial number is the last six digits of the device's MAC address, converted into decimal representation. In networks that support this feature and if the DHCP server registers this host name to a DNS server, you can access the device (through a Web browser) using the URL "http://acl_<serial number>" instead of the device's IP address. For example, if the device's MAC address is 00908f010280, the DNS name is "acl_66176".

- This section is applicable to DHCP-based provisioning of the device's IPv4 management interface (OAMP) only.
- When using DHCP to acquire an IP address, VLANs and other advanced configuration options are disabled.
- For additional DHCP parameters, see DHCP Parameters.

➢ **To enable the device as a DHCP client:**

1. Open the Network Settings page (**Setup** menu > **IP Network** tab > **Advanced** folder > **Network Settings**).
2. From the 'Enable DHCP" drop-down list, select **Enable**.

3. Click **Apply**.
4. To activate the DHCP process, reset the device.

The following shows an example of a configuration file for a Linux DHCP server (dhcpd.conf). The devices are allocated temporary IP addresses in the range 10.31.4.53 to 10.31.4.75. TFTP is assumed to be on the same computer as the DHCP server (alternatively, the "next-server" directive may be used).

```
ddns-update-style ad-hoc;
default-lease-time 60;
max-lease-time 60;
```

```
class "gateways" {
      match if(substring(hardware, 1, 3) = 00:90:8f);
}
subnet 10.31.0.0 netmask 255.255.0.0 {
      pool {
            allow members of "AudioCodes";
            range 10.31.4.53 10.31.4.75;
            filename "SIP_F6.60A.217.003.cmp -fb;device.ini";
            option routers                10.31.0.1;
            option subnet-mask            255.255.0.0;
      }
}
```

- If the DHCP server denies the use of the device's current IP address and specifies a different IP address (according to RFC 1541), the device must change its networking parameters. If this occurs while calls are in progress, they are not automatically rerouted to the new network address. Therefore, administrators are advised to configure DHCP servers to allow renewal of IP addresses.
- If the device's network cable is disconnected and then reconnected, a DHCP renewal is performed (to verify that the device is still connected to the same network). The device also includes its product name in the DHCP Option 60 Vendor Class Identifier. The DHCP server can use this product name to assign an IP address accordingly.
- After power-up, the device performs two distinct DHCP sequences. Only in the second sequence is DHCP Option 60 included. If the device is software reset (e.g., from the Web interface or SNMP), only a single DHCP sequence containing Option 60 is sent.

Provisioning from HTTP Server using DHCP Option 67

You can provision the device through HTTP using DHCP Option 67. In this setup, DHCP Option 67 informs the device of the URL address of the HTTP server from where it can download the configuration file. This provisioning method is possible only if the DHCP server allows configuration of DHCP Option values for different equipment in the network.

Upon device startup, the device sends a DHCP request. The DHCP response received from the DHCP server contains networking (e.g., IP address and DNS server) information. In addition, the response includes DHCP Option 67, which specifies the URL address of the HTTP server where the device's configuration file is located. The device then automatically downloads the configuration file from this HTTP server.

Below is an example of a configuration file (dhcpd.conf) of a Linux-based DHCP server, showing the required format of Option 67:

```
ddns-update-style ad-hoc;
default-lease-time 3600;
max-lease-time 3600;
```

```
class "AudioCodes" {
      match if(substring(hardware, 1, 3) = 00:90:8f);
}
subnet 10.31.0.0 netmask 255.255.0.0 {
      pool {
            allow members of "AudioCodes";
            range 10.31.4.53 10.31.4.75;
            option routers                10.31.0.1;
            option subnet-mask            255.255.0.0;
            option domain-name-servers    10.1.0.11;
            option bootfile-name          "INI=http://www.corp.com/master.ini";
            option dhcp-parameter-request-list 1,3,6,51,67;
      }
}
```

- The value of Option 67 must include the URL address, using the following syntax: "INI=<URL with ini file name>"
- This method is NAT-safe.

➢ **To configure the device for automatic provisioning through HTTP/S using DHCP Option 67:**

1. Enable DHCP client functionality, by configuring the following ini file parameter:

```
DHCPEnable = 1
```

2. Enable the device to include DHCP Option 67 in DHCP Option 55 (Parameter Request List) when requesting HTTP provisioning parameters from a DHCP server, using the following ini file parameter:

```
DHCPRequestTFTPParams = 1
```

3. Reset the device with a save-to-flash for your settings to take effect.

Provisioning from TFTP Server using DHCP Option 66

Provisioning the device from a third-party TFTP server is suitable when the network in which the device is deployed includes a provisioning TFTP server for all network equipment, without the capability of distinguishing between the device and other third-party devices.

Upon startup, the device checks for DHCP Option 66 in the DHCP response received from the DHCP server. If Option 66 contains a valid IP address (or FQDN) of the TFTP provisioning server, the device attempts to download through TFTP, a configuration file whose filename contains the device's MAC address (e.g., 00908f0130aa.ini).

This method loads the configuration file to the device as a one-time action. The download is repeated only if the device is manually restored to factory defaults (by pressing the hardware reset button while the Ethernet cable is not connected) and DHCP is enabled (see note below).

➢ **To configure the device for automatic provisioning through TFTP using DHCP Option 66:**

1. Enable DHCP client functionality, by configuring the following ini file parameter:

```
DHCPEnable = 1
```

2. Enable the device to include DHCP Option 66 in DHCP Option 55 (Parameter Request List) when requesting TFTP provisioning parameters from a DHCP server, using the following ini file parameter:

```
DHCPRequestTFTPParams = 1
```

3. Reset the device with a save-to-flash for your settings to take effect.

- Access to the core network through TFTP is not NAT-safe.
- The TFTP data block size (packets) when downloading a file from a TFTP server for the Automatic Update mechanism can be configured using the AUPDTftpBlockSize parameter.

Provisioning the Device using DHCP Option 160

You can provision the device using DHCP Option 160. DHCP Option 160 provides the device with the URL address of the provisioning server from where it can download its software (.cmp) and configuration (.ini) files. The URL can also include the names of the required files to download and their folder location on the server.

If you enable DHCP client functionality with DHCP Option 160, upon a device reset or power up, the device (as a DHCP client) sends a DHCP request to the DHCP server to obtain networking information (e.g., the device's IP address) and the URL address of the provisioning server.

The following syntax is supported for defining the URL and configuration/firmware filenames in DHCP Option 160 (on the DHCP server):

- <protocol>://<server IP address or hostname>
- <protocol>://<server IP address or hostname>/<software filename>
- <protocol>://<server IP address or hostname>/;<configuration filename>
- <protocol>://<server IP address or hostname>/<software filename>;<configuration filename>

The protocol can be HTTP, HTTPS, FTP, or TFTP. As shown above, the URL can include both the software and configuration filenames. In this case, they must be separated by a semicolon (;) and without spaces.

If the URL does not specify a configuration filename or the file does not exist on the provisioning server, the device requests from the server a "default" configuration file whose name includes the device's product name and MAC address (<Product><MAC>.ini, for example, "M800B00908f5b1035.ini"). If this "default" file also does not exist on the server, the device attempts to retrieve another "default" configuration file whose name includes only the device's product name (<Product>.ini, for example, "M800B.ini"). The device makes up to three attempts to download the configuration file if a failure occurs (i.e., file not exist or any other failure reason). This applies to each of the configuration files, as mentioned previously.

If the URL specifies a software file, the device makes only one attempt to download the file (even if a failure occurs). If the URL does not specify a software file, the device does not make any attempt to download a software file.

Once the device downloads the file(s), it undergoes a reset to apply the configuration and/or software. In addition, once the file(s) has been downloaded, the device ignores all future DHCP Option 160 messages. Only if the device is restored to factory defaults will it process Option 160 again (and download any required files).

➢ **To enable provisioning using DHCP Option 160:**

1. Make sure that the DHCP server is configured with the appropriate information (including the URL address of the provisioning server for Option 160).

2. Make sure that the required configuration and/or software files are located on the provisioning server.
3. Enable DHCP client functionality, as described in DHCP-based Provisioning on page 901
4. Enable the device to include DHCP Option 160 in the DHCP Parameter Request List field of the DHCP request packet that is sent to the DHCP server. Do this by loading an ini file to the device with the following parameter setting:

```
DhcpOption160Support = 1
```

1. Reset the device with a save-to-flash for your settings to take effect.

HTTP-based Provisioning

An HTTP or HTTPS server can be located in the network in which the device is deployed, storing configuration and software files for the device to download. This does not require additional servers and is NAT-safe.

For example, assume the core network HTTPS server is https://www.corp.com. A master configuration ini file can be stored on the server, for example, https://www.corp.com/gateways/master.ini. This file could point to additional ini files, Auxiliary files (e.g., call progress tones), and software files (cmp), all on the same HTTP server or different HTTP servers in the network.

The main advantage of this method is that the device can be configured to periodically check the HTTP server for file updates. HTTP(S) is not sensitive to NAT devices, enabling configuration whenever needed without on-site intervention. For additional security, the URL may contain a different port, and username and password.

The only configuration required is to preconfigure the device(s) with the URL of the initial (master) ini file. This can be done using one of the following methods:

- DHCP, as described in DHCP-based Provisioning or via TFTP at a staging warehouse. The URL is configured using the IniFileURL parameter.
- Private labeling (preconfigured during the manufacturing process).
- Manually on-site, using the RS-232 port or Web interface.

When the device is deployed at the customer site, local DHCP server provides the devices with IP addressing and DNS server information. From the URL provided in the DHCP response, the device can then contact the HTTP server at the core network and automatically download its configuration. The URL can be a simple file name or contain the device's MAC or IP address, e.g.:

- *http://corp.com/config-<MAC>.ini* - which becomes, for example, http://corp.com/config-00908f030012.ini
- *http://corp.com/<IP>/config.ini* - which becomes, for example, http://corp.com/192.168.0.7/config.ini

For more information on HTTP-based provisioning, see HTTP/S-Based Provisioning using the Automatic Update Feature.

FTP-based Provisioning

The Automatic Update feature provides limited support for FTP/FTPS connectivity. Periodic polling for updates is not possible since these protocols do not support conditional fetching (i.e., updating files only if they are changed on the server).

The only difference between FTP-based provisioning and those described in HTTP-based Provisioning is that the protocol in the URL is "ftp" (instead of "http").

Provisioning through OVOC

AudioCodes One Voice Operations Center (OVOC) server functions as a core-network provisioning server. The device's SNMP Manager should be configured with the IP address of the OVOC server, using one of the methods detailed in the previous sections. As soon as a registered device contacts OVOC through SNMP, OVOC handles all required configuration automatically, upgrading software as needed. This alternative method doesn't require additional servers at the customer premises, and is NAT-safe.

HTTP/S-Based Provisioning using the Automatic Update Feature

The Automatic Update feature can be used for automatic provisioning of the device through HTTP/S. Automatic provisioning is useful for large-scale deployment of devices. In some cases, the devices are shipped to the end customer directly from the manufacturer. In other cases, they may pass through a staging warehouse. Configuration may occur at the staging warehouse or at the end-customer premises.

The device may be preconfigured during the manufacturing process (commonly known as private labeling). Typically, a two-stage configuration process is implemented whereby initial configuration includes only basic configuration, while the final configuration is done only when the device is deployed in the live network.

If you use the IniFileURL parameter for the Automatic Update feature, do not use the Web interface to configure the device. If you do configure the device through the Web interface and save (burn) the new settings to the device's flash memory, the IniFileURL parameter is automatically set to 0 and Automatic Updates is consequently disabled. To enable Automatic Updates again, you need to re-load the ini file (using the Web interface or BootP) with the correct IniFileURL settings. As a safeguard to an unintended save-to-flash when resetting the device, if the device is configured for Automatic Updates, the 'Burn To FLASH' field under the Reset Configuration group in the Web interface's Maintenance Actions page is automatically set to No by default.

- For a description of all the Automatic Update parameters, see Automatic Update Parameters or refer to the *CLI Reference Guide*.
- For additional security, use HTTPS or FTPS. The device supports HTTPS (RFC 2818) and FTPS using the AUTH TLS method <draft-murray-auth-ftp-ssl-16>.

Files Provisioned by Automatic Update

You can use the Automatic Update feature to update the device with any of the following files:

- Software file (*.cmp)*
- License Key file
- Auxiliary files (e.g., Dial Plan file, SBC User Information file, and Call Progress Tones file)
- TLS security files (trusted root certificate file, TLS certificate file, and private key)
- Web GUI file (favicon file)
- Configuration file:
  - **ini File:** Contains only ini file parameters and configures all the device's functionality.
  - **CLI Script File:** Contains only CLI commands and configures all the device's functionalities (except commands such as show, debug or copy). The file updates the device's configuration only according to the configuration settings in the file. The device's existing configuration settings (not included in the file) are retained. The device does not undergo a reset and therefore, this file typically contains configuration settings that do not require a device reset. If a reset is required, for example, to apply certain settings, you must include the following CLI command (root level) at the end of the file:

    ```
    # reload if-needed
    ```

    To configure the URL of the server where the file is located, use the CliScriptURL ini file parameter or CLI command, configure system > automatic-update > cli-script <URL>.
  - **Startup Script File:** Contains only CLI commands and configures all the device's functionality (except commands such as show, debug or copy). The file updates the device's configuration according to the configuration settings in the file and sets all other parameters that are not included in the file to factory defaults. The file causes two device resets to apply the settings. Therefore, the file typically contains the Automatic Update settings and other configuration settings that require a device reset. The URL of the server where this file is located is configured by the CLIStartupScriptURL ini file parameter or CLI command, configure system > automatic-update > startup-script <URL>.

- You can use any filename extension for the CLI script files.
- The CLI Startup Script file is not supported when the device operates in HA mode.

File Location for Automatic Update

The files for updating the device can be stored on any standard Web (HTTP/S), TFTP, or FTP server. The files can be loaded periodically to the device using HTTP/S, TFTP, or FTP. This mechanism can be used even when the device is installed behind NAT and firewalls. The Automatic Update feature is done per file and configured by specifying the file name and URL address of the provisioning server where the file is located. If the device needs to authenticate itself with the server, you can use the same parameters to configure the authentication username and password (for more information, see Access Authentication with HTTP Server on page 914). For a description of the parameters for configuring the URLs of the servers of the files, see Automatic Update Parameters.

Below are examples for configuring the file names and their URLs for Automatic Update:

- ini File:

```
IniFileURL = 'http://www.corp.com/configuration.ini'
CptFileURL = 'http://www.corp.com/call_progress.dat'
AutoCmpFileUrl = 'http://www.corp.com/SIP_F7.20A.008.cmp
FeatureKeyURL = 'https://www.company.com/License_Key.txt'
```

- CLI:

```
# configure system
(config-system)# automatic update
(automatic-update)# cli-script https://company.com/cli/<MAC>
(automatic-update)# startup-script https://company.com/startup/<MAC>
(automatic-update)# ini-file http://www.company.com/configuration.ini
(automatic-update)# call-progress-tones http://www.company.com/call_
progress.dat
(automatic-update)# feature-key http://www.company.com/License_Key.txt
(automatic-update)# auto-firmware http://www.company.com/SIP_
F7.20A.008.cmp
```

- For configuration files, the file name in the URL can automatically contain the device's MAC address for enabling the device to download a file unique to the device. For more information, see MAC Address Placeholder in Configuration File Name.
- When using the [IniFileURL] parameter, parameters not included in the file are restored to default settings. If you want to keep the settings of these parameters, use the [IncrementalIniFileURL] parameter instead.

MAC Address Placeholder in Configuration File Name

You can configure the file name of the configuration file in the URL to automatically include the MAC address of the device, as described in File Location for Automatic Update

```
IniFileURL = 'https://www.company.com/config_<MAC>.ini'
(automatic-update)# cli-script https://company.com/files/cli_script_<MAC>.txt
(automatic-update)# startup-script https://company.com/files/startup_<MAC>.txt
```

The device automatically replaces the string with its hardware MAC address, resulting in a file name request that contains the device's MAC address, for example, config_00908F033512.ini or startup_00908F033512.txt. Therefore, you can configure all the devices with the same URL and file name.

If you write the MAC address placeholder string in lowercase (i.e., "<mac>"), the device adds the MAC address in lowercase to the file name (e.g., config_<mac>.ini results in config_00908**f**053736**e**); if in uppercase (i.e., "<MAC>"), the device adds the MAC address in uppercase to the file name (e.g., config_<MAC>.ini results in config_00908**F**053736**E**).

Although the device connects to the network through its WAN port, the file must be named with the device's **LAN** MAC address.

File Template for Automatic Provisioning

To facilitate automatic provisioning setup, you can use a single template to define the files to download during automatic provisioning. The template uses special keywords to denote the different file types to download and in the URL address of the provisioning server it uses a placeholder for the file names which is replaced by hardcoded file names and extensions according to file type, as described in more detail below.

- Unlike the parameters that define specific URLs for Auxiliary files (e.g., CptFileURL), the file template feature always retains the URLs after each automatic update process. Therefore, with the file template the device always attempts to download the files upon each automatic update process.
- If you configure a parameter used to define a URL for a specific file (e.g., CptFileURL), the settings of the TemplateUrl parameter is ignored for the specific file type (e.g., CPT file).
- Additional placeholders can be used in the file name in the URL, for example, <MAC> for MAC address (see MAC Address Placeholder in Configuration File Name).

➢ To use a file template for automatic provisioning:

1. Define the file **types** to download by the file template, using the AupdFilesList parameter. Use the keywords listed in the table below to specify each file type. For example, to specify ini, License Key, and CPT files:

   - ini File:

     ```
     AupdFilesList = 'ini', 'fk', 'cpt'
     ```

   - CLI:

     ```
     # configure system
     (config-system)# automatic update
     (automatic-update)# template-files-list ini,fk,cpt
     ```

2. Define the URL address of the provisioning server on which the files (specified in Step 1) are located for download, using the TemplateUrl parameter. When you configure the URL, you must include the file type placeholder, "<FILE>", which represents the file name. For each file type specified in Step 1, the device sends an HTTP request to the server, where the placeholder in the URL is replaced with the filename and extension, as listed in the below table. For example, if you configure the AupdFilesList parameter as in Step 1 and the TemplateUrl parameter to:

   - ini File:

     ```
     TemplateUrl = 'http://10.8.8.20/Site1_<FILE>'
     ```

   - CLI:

     ```
     # configure system
     (config-system)# automatic update
     (automatic-update)# template-url http://10.8.8.20/Site1_<FILE>
     ```

   The device sends HTTP requests to the following URLs:

- http://10.8.8.20/Site1_**device.ini**
- http://10.8.8.20/Site1_**fk.ini**
- http://10.8.8.20/Site1_**cpt.data**

3. Place the files to download on the provisioning server. Make sure that their file names and extensions are based on the hardcoded string values specific to the file type for the <FILE> placeholder (e.g., "Site1_device.ini" for the ini file), as shown in the table below.

Table 39-1: File Template Keywords and Placeholder Values per File Type

File Type	Keywords for Template File	Value Replacing <FILE> Placeholder
ini file	ini	device.ini
CLI Script file	cli	cliScript.txt
CLI Startup Script file	clis	cliStartupScript.txt
CMP file based on timestamp	acmp	autoFirmware.cmp
User Information file	usrinf	userInfo.txt
CMP file	cmp	firmware.cmp
License Key file	fk	fk.ini
Call Progress Tone (CPT) file	cpt	cpt.dat
Prerecorded Tones (PRT) file	prt	prt.dat
Dial Plan file	dpln	dialPlan.dat
Answering Machine Detection (AMD) file	amd	amd.dat
TLS Private Key file	sslp	pkey.pem pkey<ID>.pem (for multi-certificate system)
TLS Root Certificate file	sslr	root.pem root<ID>.pem (for multi-certificate system)
TLS Certificate file	sslc	cert.pem cert<ID>.pem (for multi-certificate system)

Triggers for Automatic Update

The Automatic Update feature can be triggered by the following:

- Upon device startup (reset or power up). To disable this trigger, run the following CLI command:

```
(config-system)# automatic-update
(automatic-update)# run-on-reboot off
```

- Periodically:
  - Specified time of day (e.g., 18:00), configured by the ini file parameter [AutoUpdatePredefinedTime] or CLI command configure system > automatic-update > predefined-time. You can configure (using the [AutoUpdatePredefinedRandomTime] parameter) an interval from the specified time in which the automatic update is randomly triggered. This is useful for reducing load on the provisioning server when you have deployed multiple devices that are implementing the Automatic Update feature. For example, if you configure [AutoUpdatePredefinedTime] to 18:00 and [AutoUpdatePredefinedRandomTime] to 300 seconds (i.e., 5 min.), the automatic update process is randomly triggered anywhere between 18:00 and 18:05.
  - Interval between Automatic Updates (e.g., every 60 minutes), configured by the ini file parameter [AutoUpdateFrequencySeconds] or CLI command configure system > automatic-update > update-frequency-sec.

Configure either [AutoUpdatePredefinedTime] or [AutoUpdateFreqencySeconds]; not both. When configuring one of the parameters, make sure that the other parameter is at its default value (i.e., disabled).

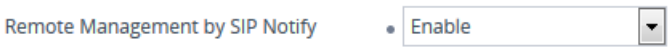
- Centralized provisioning server request:
  - Upon receipt of an SNMP request from the provisioning server.
  - Upon receipt of a special SIP NOTIFY message from the provisioning server. The NOTIFY message includes an Event header with the AudioCodes proprietary value, "check-sync;reboot=false", as shown in the example below:

```
NOTIFY sip:<user>@<dsthost> SIP/2.0
To: sip:<user>@<dsthost>
From: sip:sipsak@<srchost>
CSeq: 10 NOTIFY
Call-ID: 1234@<srchost>
Event: check-sync;reboot=false
```

To enable the feature:

i. Open the SIP Definitions General Settings page (**Setup** menu > **Signaling & Media** tab > **SIP Definitions** folder > **SIP Definitions General Settings**).

ii. From the 'Remote Management by SIP Notify' (EnableSIPRemoteReset) drop-down list, select **Enable**:



iii. Click **Apply**.

To enable through CLI: configure voip > sip-definition advanced-settings > sip-remote-reset.

Applying Downloaded ini File after Graceful Timeout

If you use the Automatic Update feature for updating the device's configuration from an ini file, you can configure the device to gracefully lock itself before applying the settings of the ini file. When the Automatic Update feature is triggered (for example, by a device reset) and the device downloads the ini file from the remote provisioning server, the graceful timeout begins. During this period, the device does not accept any new calls, allowing only existing calls to continue until the timeout expires. If all existing calls end before the timeout expires, the device applies the configuration of the downloaded ini file. If there are still existing calls when the timeout expires, the device terminates the calls, and then applies the configuration of the downloaded ini file.

➢ **To configure graceful timeout for automatic update of ini file:**

1. In the ini file used for enabling and configuring the device for Automatic Update, include the following parameters with the other parameters (such as IniFileURL) relating to Automatic Update setup:

```
...
AupdGracefulShutdown=1
AdminStateLockControl=<Graceful Timeout>
...
```

2. Load the ini file to the device.

Access Authentication with HTTP Server

You can configure the device to authenticate itself with the HTTP/S server storing the files that you want to download for the Automatic Update mechanism. The device authenticates itself by providing the HTTP/S server with its authentication username and password. The credentials are used for both Digest access authentication (MD5 cryptographic hashing) and the non-secured Basic access authentication method.

When configuring the URL of the server with the name of the file that you want downloaded, you can also include the username and password in the format "username:password" (without quotation marks), as shown in the example below for the software file (.cmp):

- ini file:

```
AutoCmpFileUrl = 'https://JoeD:1234@10.1.1.1/mysw.cmp'
```

- CLI:

```
# configure system
(config-system)# automatic update
(automatic-update)# auto-firmware https://JoeD:1234@10.1.1.1/mysw.cmp
```

If you have not included the username and password in the parameters used for configuring the URL of the server with the name of the file that you want downloaded, the device uses the username and password that you configured for the ini file parameter [AUPDUserPassword] or CLI command `configure system > automatic-update > credentials`.

The password cannot be configured with wide characters (for example, Chinese characters).

Querying Provisioning Server for Updated Files

Each time the Automatic Update feature is triggered, for each file and its configured URL the device does the following:

1. If you have configured the device to authenticate itself to the HTTP/S server for secure access, the device sends the access authentication username and password to the HTTP/S server (for more information, see Access Authentication with HTTP Server). If authentication succeeds, Step 2 occurs.
2. The device establishes an HTTP/S connection with the URL host (provisioning server). If the connection is HTTPS, the device verifies the certificate of the provisioning server, and presents its own certificate if requested by the server.
3. The device queries the provisioning server for the requested file by sending an HTTP Get request. This request contains the HTTP User-Agent Header, which identifies the device to the provisioning server. By default, the header includes the device's model name, MAC address, and currently installed software and configuration versions. Based on its own dynamic applications for logic decision making, the provisioning server uses this information to check if it has relevant files available for the device and determines which files must be downloaded (working in conjunction with the HTTP If-Modified-Since header, described further on in this section).

   You can configure the information sent in the User-Agent header, using the [AupdHttpUserAgent] parameter or CLI command, `configure system > http-`

`user-agent`. The information can include any user-defined string or the following supported string variable tags (case-sensitive):

- **<NAME>:** product name, according to the installed License Key
- **<MAC>:** device's MAC address
- **<VER>:** software version currently installed on the device, e.g., "7.00.200.001"
- **<CONF>:** configuration version, as configured by the ini file parameter, [INIFileVersion] or CLI command, `configuration-version`

The device automatically populates these tag variables with actual values in the sent header. By default, the device sends the following in the User-Agent header:

```
User-Agent: Mozilla/4.0 (compatible; AudioCodes;
<NAME>;<VER>;<MAC>;<CONF>)
```

For example, if you configure [AupdHttpUserAgent] to "MyWorld-<NAME>;<VER> (<MAC>)", the device sends the following User-Agent header:

```
User-Agent: MyWorld-Mediant;7.00.200.001(00908F1DD0D3)
```

If you configure the [AupdHttpUserAgent] parameter with the <CONF> variable tag, you must reset the device with a save-to-flash for your settings to take effect.

4. If the provisioning server has relevant files available for the device, the following occurs, depending on file type and configuration:

   - **File Download upon each Automatic Update process:** This is applicable to software (.cmp) and configuration files.In the sent HTTP Get request, the device uses the HTTP If-Modified-Since header to determine whether to download these files. The header contains the date and time (timestamp) of when the device last downloaded the file from the specific URL. This date and time is regardless of whether the file was installed or not on the device. An example of an If-Modified-Since header is shown below:

     ```
     If-Modified-Since: Mon, 1 January 2014 19:43:31 GMT
     ```

     If the file on the provisioning server was unchanged (not modified) since the date and time specified in the header, the server replies with an HTTP 304 response and the file is not downloaded. If the file was modified, the provisioning server sends an HTTP 200 OK response with the file in the body of the HTTP response. The device downloads the file and compares the version of the file with the currently installed version on its flash memory. If the downloaded file is of a later version, the device installs it after the device resets (which is only done after the device completes all file downloads); otherwise, the device does not reset and does not install the file.

     To enable the automatic software (.cmp) file download method based on this timestamp method, use the [AutoCmpFileUrl] parameter or CLI command

`configure system > automatic-update > auto-firmware <URL>`. The device uses the same configured URL to download the .cmp file for each subsequent Automatic Update process.

You can also enable the device to run a CRC on the downloaded configuration file to determine whether the file has changed in comparison to the previously downloaded file. Depending on the CRC result, the device can install or discard the downloaded file. For more information, see Cyclic Redundancy Check on Downloaded Configuration Files.

- When this method is used, there is typically no need for the provisioning server to check the device's current firmware version using the HTTP-User-Agent header.
- The Automatic Update feature assumes that the Web server conforms to the HTTP standard. If the Web server ignores the If-Modified-Since header or doesn't provide the current date and time during the HTTP 200 OK response, the device may reset itself repeatedly. To overcome this problem, modify the update frequency (interval), using the [AutoUpdateFreqencySeconds] parameter or CLI command `configure system > automatic update > update-frequency-sec`.

- **One-time File Download:** This is applicable to software (.cmp) and Auxiliary (e.g., License Key, CPT and Dial Plan) files. The device downloads these files only **once**, regardless of how many times the device may repeat the Automatic Update process. Once they are downloaded, the device discards their configured URLs. To update these files again, you need to configure their URL addresses and filenames again. Below is an example of how to configure URLs for some of these files:

  Auxiliary Files:

  - ini:

    ```
    CptFileURL = 'https://www.company.com/call_progress.dat'
    FeatureKeyURL = 'https://www.company.com/License_Key.txt'
    ```

  - CLI:

    ```
    (config-system)# automatic-update
    (automatic-update)# call-progress-tones
    http://www.company.com/call_progress.dat
    (automatic-update)# tls-root-cert https://company.com/root.pem
    ```

  Software (.cmp) File:

  - ini:

    ```
    CmpFileUrl = 'https://www.company.com/device/7.20A.258.980.cmp'
    ```

  - CLI:

```
(config-system)# automatic-update
(automatic-update)# firmware
https://www.company.com/device/7.20A.258.980.cmp
```

- For one-time file download, the HTTP Get request sent by the device does not include the If-Modified-Since header. Instead, the HTTP-User-Agent header can be used in the HTTP Get request to determine whether firmware update is required.
- When downloading TLS certificate files, it is recommended to use HTTPS with mutual authentication for secure transfer of the TLS Private Key.
- For devices in HA mode, the License Key is applied to both active and redundant units.

5. If the device receives an HTTP 301/302/303 redirect response from the provisioning server, it establishes a connection with the new server at the redirect URL and re-sends the HTTP Get request.

File Download Sequence

Whenever the Automatic Update feature is triggered (see Triggers for Automatic Update), the device attempts to download the files (if available) from the configured URLs in the following order:

1. ini file (.ini)
2. CLI Script file (.txt)
3. CLI Startup Script file (.txt)
4. Periodic software file (.cmp) download
5. One-time software file (.cmp) download
6. Auxiliary file(s)

The following files automatically instruct the device to reset:

- CLI Startup Script file
- Periodic software file (.cmp)
- One-time software file (.cmp)

When multiple files requiring a reset are downloaded, the device resets only **after** it has downloaded and installed **all** the files. However, you can explicitly instruct the device to immediately reset for the following files:

- ini file: Use the [ResetNow] in file parameter
- CLI Script file: Use the `reload if-needed` CLI command

If you use the [ResetNow] parameter in an ini file for periodic automatic provisioning with non-HTTP (e.g., TFTP) and without CRC, the device resets after every file download. Therefore, use the parameter with caution and only if necessary for your deployment requirements.

- For ini file downloads, by default, parameters not included in the file are set to defaults. To retain the current settings of these parameters, set the SetDefaultOnINIFileProcess parameter to 0.
- The CLI Startup Script file is not supported when the device operates in HA mode.
- If you have configured one-time software file (.cmp) download (configured by the [CmpFileURL] parameter or CLI command `configure system > automatic-update > firmware`), the device will only apply the file if one-time software updates are enabled. This is disabled by default to prevent unintentional software upgrades. To enable one-time software upgrades, set the [AutoUpdateCmpFile] parameter to [1] or CLI command, `configure system > automatic-update > update-firmware on`.
- If you need to update the device's software and configuration, it is recommended to first update the software. This is because the current ("old") software (before the upgrade) may not be compatible with the new configuration. However, if both files are available for download on the provisioning server(s), the device first downloads and applies the new configuration, and only then does it download and install the new software. Therefore, this is a very important issue to take into consideration.
- If more than one file needs to be updated:
  - ✓ CLI Script and cmp: The device downloads and applies the CLI Script file on the currently ("old") installed software version. It then downloads and installs the cmp file with a reset. Therefore, the CLI Script file **MUST** have configuration compatible with the "old" software version.
  - ✓ CLI Startup Script and cmp: The device downloads both files, resets, applies the new cmp, and then applies the configuration from the Startup Script file on the new software version.
  - ✓ CLI Script and Startup Script: The device downloads and applies both files; but the Startup Script file overwrites all the configuration of the CLI Script file

Cyclic Redundancy Check on Downloaded Configuration Files

You can enable the device to perform cyclic redundancy checks (CRC) on downloaded configuration files during the Automatic Update process. The CRC checks whether the content (raw data) of the downloaded file is different to the content of the previously downloaded file from the previous Automatic Update process. The device compares the CRC check value (code) result with the check value of the previously downloaded file. If the check values are identical, it indicates that the file has no new configuration settings, and the device discards the file. If the check values are different, it indicates that the downloaded file is different (i.e., includes updates), and the device installs the downloaded file and applies the new configuration settings.

CRC is useful, for example, when the service provider replaces a file, on the provisioning server, with another file whose contents are the same. When the device sends an HTTP Get request during the Automatic Update process, the provisioning server sends the new file to the device. This occurs as the timestamp between the previously downloaded file and this new file is different (determined by the HTTP If-Modified-Since header in the Get request). Therefore, the CRC feature can be used to prevent the device from installing such files.

For enabling CRC, use the ini file parameter AUPDCheckIfIniChanged or CLI command, configure system > automatic-update > crc-check regular. By default, CRC is disabled. For more information on the parameter, see Automatic Update Parameters.

Automatic Update Configuration Examples

This section provides a few examples on configuring the Automatic Update feature.

Automatic Update for Single Device

This simple example describes how to configure the Automatic Update feature for updating a single device. In this example, the device queries the provisioning server for software, configuration and Auxiliary files every 24 hours.

➢ **To set up Automatic Provisioning for single device (example):**

1. Set up an HTTP Web server (e.g., http://www.company.com) and place all the required configuration files on this server.
2. Configure the device with the IP address of the DNS server for resolving the domain name (e.g., http://www.company.com) that is used in the URL of the provisioning server. You configure this in the IP Interfaces table:
   - ini File:

```
[ InterfaceTable ]
FORMAT InterfaceTable_Index = InterfaceTable_ApplicationTypes,
InterfaceTable_InterfaceMode, InterfaceTable_IPAddress,
InterfaceTable_PrefixLength, InterfaceTable_Gateway, InterfaceTable_
VlanID, InterfaceTable_InterfaceName, InterfaceTable_
PrimaryDNSServerIPAddress, InterfaceTable_
SecondaryDNSServerIPAddress, InterfaceTable_
UnderlyingDevice;InterfaceTable 0 = 6, 10, 10.15.7.95, 16, 10.15.0.1, 1,
"Voice", 80.179.52.100, 0.0.0.0, "vlan 1";
[ \InterfaceTable ]
```

   - CLI:

```
# configure network
(config-network)# interface network-if 0
(network-if-0)# primary-dns 80.179.52.100
```

3. Configure the device with the following Automatic Update settings:

   a. Automatic Update is done every 24 hours (86,400 seconds):

      - ini File:

        ```
        AutoUpdateFreqencySeconds = 86400
        ```

      - CLI:

        ```
        # configure system
        (config-system)# automatic update
        (automatic-update)# update-frequency-sec  86400
        ```

   b. Automatic Update of software file (.cmp):

      - ini File:

        ```
        AutoCmpFileUrl =  'https://www.company.com/sw.cmp'
        ```

      - CLI:

        ```
        # configure system
        (config-system)# automatic update
        (automatic-update)# auto-firmware
        'http://www.company.com/sw.cmp'
        ```

   c. Automatic Update of Call Progress Tone file:

      - ini File:

        ```
        CptFileURL = 'https://www.company.com/call_progress.dat'
        ```

      - CLI:

        ```
        # configure system
        (config-system)# automatic update
        (automatic-update)# call-progress-tones
        'http://www.company.com/call_progress.dat'
        ```

   d. Automatic Update of ini configuration file:

- ini File:

```
IniFileURL = 'https://www.company.com/config.ini'
```

- CLI:

```
# configure system
(config-system)# automatic update
(automatic-update)# ini-file 'http://www.company.com/config.ini'
```

e. Enable Cyclical Redundancy Check (CRC) on downloaded ini file:

- ini File:

```
AUPDCheckIfIniChanged = 1
```

- CLI:

```
# configure system
(config-system)# automatic update
(automatic-update)# crc-check regular
```

4. Power down and then power up the device.

Automatic Update from Remote Servers

This example describes how to configure the Automatic Update feature where files are stored and downloaded from different file server types. The example scenario includes the following:

- FTPS server at ftpserver.corp.com for storing the License Key file. The login credentials to the server are username "root" and password "wheel".
- HTTP server at www.company.com for storing the configuration file.
- DNS server at 80.179.52.100 for resolving the domain names of the provisioning servers (FTPS and HTTP).

➢ **To set up Automatic Provisioning for files stored on different server types (example):**

1. **License Key file:**

a. Set up an FTPS server and copy the License Key file to the server.

b. Configure the device with the URL path of the License Key file:

- ini File:

```
FeatureKeyURL = 'ftps://root:wheel@ftpserver.corp.com/license_
key.txt'
```

- CLI:

```
# configure system
(config-system)# automatic update
(automatic-update)# feature-key
'ftps://root:wheel@ftpserver.corp.com/license_key.txt'
```

2. **Software (.cmp) and ini files:**
   a. Set up an HTTP Web server and copy the .cmp and configuration files to the server.
   b. Configure the device with the URL paths of the .cmp and ini files:
      - ini File:

```
AutoCmpFileUrl =  'http://www.company.com/device/sw.cmp'
IniFileURL = 'http://www.company.com/device/inifile.ini'
```

      - CLI:

```
# configure system
(config-system)# automatic update

(automatic-update)# auto-firmware
'http://www.company.com/sw.cmp'
(automatic-update)# startup-script https://company.com/files/startup_
script.txt
```

3. Configure the device with the IP address of the DNS server for resolving the domain names of the FTPS and HTTP servers:

```
[ InterfaceTable ]
FORMAT InterfaceTable_Index = InterfaceTable_ApplicationTypes,
InterfaceTable_InterfaceMode, InterfaceTable_IPAddress, InterfaceTable_
PrefixLength, InterfaceTable_Gateway, InterfaceTable_VlanID,
InterfaceTable_InterfaceName, InterfaceTable_PrimaryDNSServerIPAddress,
InterfaceTable_SecondaryDNSServerIPAddress, InterfaceTable_
UnderlyingDevice;
InterfaceTable 0 = 6, 10, 10.15.7.95, 16, 10.15.0.1, 1, "Voice", 80.179.52.100,
0.0.0.0, "vlan 1";
[ \InterfaceTable ]
```

4. Configure the device to perform the Automatic Update process daily at 03:00 (3 a.m):
   - ini File:

     ```
     AutoUpdatePredefinedTime = '03:00'
     ```

   - CLI:

     ```
     # configure system
     (config-system)# automatic update
     (automatic-update)# predefined-time  03:00
     ```

Automatic Update for Mass Deployment

This example describes how to configure the Automatic Update feature for updating multiple devices (i.e., mass deployment) using an HTTP provisioning server. In this example, all the devices are configured to download the same "master" configuration file. This file serves as the configuration template and instructs the devices which files to download and how often to perform the Automatic Update process. In addition, the master file also instructs each device to download an ini configuration file whose file name contains the MAC address of the device.

The example scenario is as follows:

- All devices download a "master" configuration file that contains the following:
  - Common configuration shared by all devices.
  - MAC address placeholder in filename (see MAC Address Placeholder in Configuration File Name on page 910)

> **Note:** When provisioning the device based on MAC address (e.g., startup-script http://-company.com/startup/<MAC>), although the device connects to the network with the WAN port, the name of the file on the provisioning server must use the **LAN** MAC address.

- Device queries the provisioning server daily at 24:00 (midnight) for software, configuration and Auxiliary files.
- HTTP-based provisioning server at www.company.com for storing the files.
- DNS server at 80.179.52.100 for resolving the domain name of the provisioning server.

➢ **To set up automatic provisioning for mass provisioning (example):**

1. Create a "master" configuration file template named "master_configuration.ini" or "master_startup.txt" with the following settings:
   - Common configuration for all devices:
     - ini file:

```
AutoUpdatePredefinedTime = '24:00'
CptFileURL = 'https://www.company.com/call_progress.dat'
AutoCmpFileUrl = 'https://www.company.com/sw.cmp'
```

- CLI:

```
# configure system
(config-system)# automatic update
 (automatic-update)# predefined-time 24:00
 (automatic-update)# call-progress-tones
https://www.company.com/call_progress.dat
(automatic-update)# auto-firmware
https://www.company.com/sw.cmp
```

- Configuration per device based on MAC address:
  - ini file:

```
IniFileURL = 'http://www.company.com/config_<MAC>.ini'
```

  - CLI:

```
# configure system
 (config-system)# automatic update
(automatic-update)# cli-script https://company.com/files/cli_script_
<MAC>.txt
(automatic-update)# ini-file http://www.company.com/config_
<MAC>.ini
```

2. Copy the master configuration file that you created in Step 1 as well as the CPT and .cmp files to the HTTP-based provisioning server.
3. Configure **each** device with the following:
   a. URL of the master configuration file:
      - ini File:

```
IniFileURL = 'http://www.company.com/master_configuration.ini'
```

      - CLI:

```
# configure system
(config-system)# automatic update
(automatic-update)# ini-file http://www.company.com/master_
```

```
configuration.ini
(automatic-update)# cli-script https://company.com/files/master_
startup.txt
```

b. Configure the device with the IP address of the DNS server for resolving the domain name (e.g., http://www.company.com) that is used in the URL for the provisioning server. This is done in the IP Interfaces table:

- ini File:

```
[ InterfaceTable ]
FORMAT InterfaceTable_Index = InterfaceTable_ApplicationTypes,
InterfaceTable_InterfaceMode, InterfaceTable_IPAddress,
InterfaceTable_PrefixLength, InterfaceTable_Gateway,
InterfaceTable_VlanID, InterfaceTable_InterfaceName,
InterfaceTable_PrimaryDNSServerIPAddress, InterfaceTable_
SecondaryDNSServerIPAddress, InterfaceTable_UnderlyingDevice;
InterfaceTable 0 = 6, 10, 10.15.7.95, 16, 10.15.0.1, 1, "Voice",
80.179.52.100, 0.0.0.0, "vlan 1";
[ \InterfaceTable ]
```

- CLI:

```
# configure network
(config-network)# interface network-if 0
(network-if-0)# primary-dns 80.179.52.100
```

4. Power down and then power up the device.

40 SBC Configuration Wizard

The SBC Configuration Wizard provides you with a quick-and-easy method for initial configuration of your device. The SBC Configuration Wizard guides you through a sequence of pages, assisting you in defining your device. Once the wizard is complete, your device is sufficiently configured to successfully process and route calls in your deployment.

The SBC Configuration Wizard is based on partially and fully tested interoperability setups between the device and a wide range of vendors, including SIP trunking providers, IP PBXs, and contact centers. Once you have selected the involved vendors and defined basic settings in the SBC Configuration Wizard, it then generates a configuration file based on the matching interoperability configuration template. Alternatively, instead of basing your configuration on specific vendors, you can use the SBC Configuration Wizard to generate a configuration file based on a generic template for commonly used setups. In such cases, you may later need to manually fine-tune your configuration to suit your setup needs.

The SBC Configuration Wizard can automatically load the generated configuration (with a reset) to the device, or you can simply save the generated configuration file to a folder on your PC and then load the file to the device at a later stage.

The generated configuration is a good starting point to enable the successful establishment of basic calls. For complete device configuration, you may need to manually configure additional functionality. For example, you may need to configure security settings (e.g., firewalls and IDS) to ensure that the device is protected from malicious activity and DoS attacks.

For AudioCodes' full interoperability list, visit AudioCodes website at https://www.audiocodes.com/partners/sbc-interoperability-list.

- When the device operates in High-Availability (HA) mode, the SBC Configuration Wizard is not supported (and not accessible from the Web interface).
- When the device is configured for WebSocket tunneling with OVOC (see Configuring WebSocket Tunnel with OVOC on page 91), the SBC Configuration Wizard is not supported (and not accessible from the Web interface).
- When the SBC Configuration Wizard applies the configuration template to the device, all parameters configured by the SBC Configuration Wizard overwrite the device's existing configuration of those parameter. Parameters not configured by the SBC Configuration Wizard are restored to factory defaults, except basic device settings such as management users (Web and CLI). Some of these basic settings also appear in the SBC Configuration Wizard and their fields are automatically populated with their current settings; if you do modify them in the SBC Configuration Wizard, their new settings are used.
- On some wizard pages, the availability of certain fields depends on the selected application.

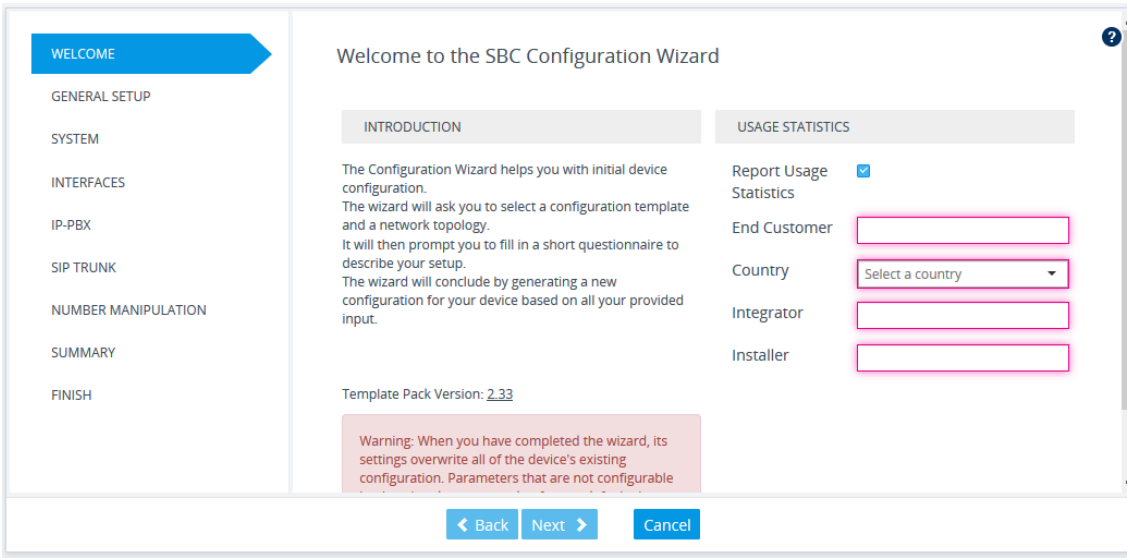
Starting the SBC Configuration Wizard

The following procedure describes how to start the SBC Configuration Wizard. Throughout the wizard, you can get help on the current wizard page, by clicking the

icon, located on the top-right of the page.

➢ To start the SBC Configuration Wizard:

1. Access the SBC Configuration Wizard's Welcome page:
   - **Toolbar:** Click **Actions**, and then from the drop-down list, choose **Configuration Wizard**.
   - **Navigation Tree: Setup** menu > **Administration** tab > **Maintenance** folder > **Configuration Wizard**.



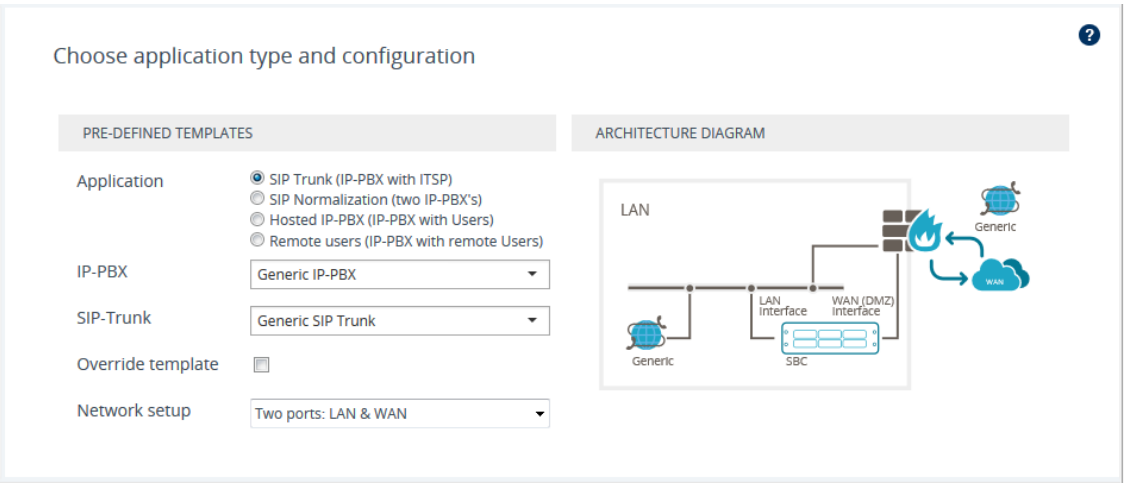
2. If desired, the SBC Configuration Wizard allows you to share usage statistics with AudioCodes to help us improve our software. To agree, select the 'Report Usage Statistics' check box, and then fill in the subsequent mandatory fields; otherwise, clear the 'Report Usage Statistics' check box.
3. The version of the template pack currently installed on the device is displayed in the 'Template Pack Version' field. The template pack contains the interoperability configuration templates available on the SBC Configuration Wizard. If the template pack is the latest, "Template pack is up to date" is displayed below the field. If the template pack is not the latest version, you can update it by clicking the **Update from Remote Server** button (see below note). Alternatively, if you have received a template pack file from the sales representative of your purchased device, you can install it on the device using the Auxiliary Files page (see Loading Auxiliary Files). If you click the version number, the Template Pack Version History window appears, displaying the updates for the current and previous versions.

The device checks with AudioCodes server if it has the latest template pack only if it has Internet connectivity. Therefore, the **Update from Remote Server** button is displayed only if the device has connectivity to the Internet and has subsequently found that the template pack is not the latest.

4. Click **Next**; the General Setup page appears (see General Setup Page).

General Setup Page

The General Setup wizard page defines the network topology of the device, which includes the application (e.g., SIP trunk) and the involved third-party vendors, configuration template based on the selected vendor interoperability, and physical network (ports). The wizard displays an illustration of the basic architecture according to your chosen setup.



➢ To define the general setup:

1. Select one of the following 'Application' options:

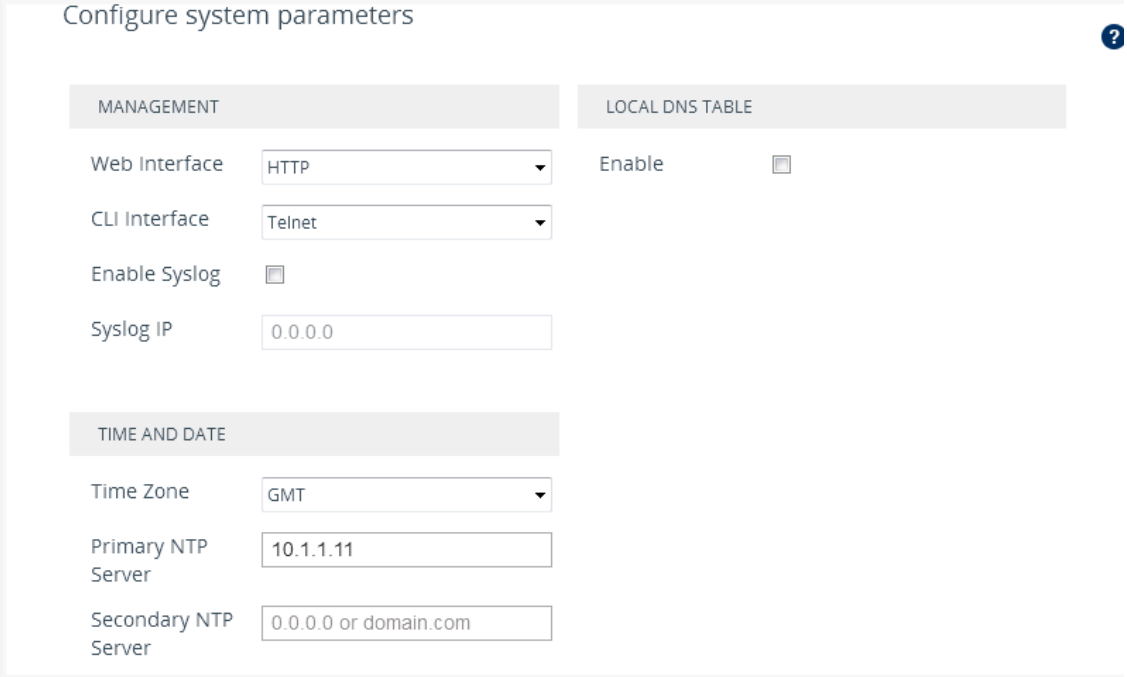
   - **SIP Trunk (IP-PBX with SIP Trunk):** The device connects the Enterprise IP PBX with the Internet Telephony Service Provider (ITSP) or SIP Trunk Provider. The IP PBX resides on the Enterprise LAN, while the ITSP resides on the WAN. Only calls between the IP PBX and ITSP traverse the device. Calls between Enterprise phones, users and certain SIP messages (REGISTER, SUBSCRIBE and NOTIFY) are managed locally by the IP PBX and do not traverse the device.
   - **SIP Normalization (two IP-PBX's):** The device performs SIP "normalization" of traffic between two IP PBXs. The first IP PBX resides on the Enterprise LAN (co-located with the device) and the second IP PBX resides on the WAN (or at another branch site). Only calls between the IP PBXs traverse the device. Calls between the phones and users of the same IP PBX, and certain SIP messages (REGISTER, SUBSCRIBE and NOTIFY) are managed locally by each IP PBX and do not traverse the device.
   - **Hosted IP-PBX (IP-PBX with Users):** The device connects phones (users) with the "hosted" IP PBX. The users reside on the Enterprise LAN (co-located with the device) and the IP PBX resides on the WAN (or at the datacenter). All traffic between users and IP PBX traverse the device, including SIP REGISTER, SUBSCRIBE and NOTIFY messages.
   - **Remote Users (IP-PBX with Remote Users):** The device connects remote phones (users) with the "local" IP PBX (additional IP PBX servers can be configured). The IP PBX resides on the Enterprise LAN (co-located with the device) and the remote users reside

on the WAN (or at the datacenter). All traffic between users and IP PBX traverse the device, including SIP REGISTER, SUBSCRIBE and NOTIFY messages.

2. If you selected the **SIP Trunk** application in Step 1, do the following:
   a. From the 'IP-PBX' drop-down list, select the IP PBX model. If the model is not listed, select **Generic IP-PBX**.
   b. From the 'SIP-Trunk' drop-down list, select the SIP trunk provider. If the provider is not listed, select **Generic SIP Trunk**.
   c. To generate a configuration template based on the individual properties of the selected IP PBX and SIP Trunk, instead of using the existing template for the specific combination, select the 'Override template' check box.
3. If you selected any application except **SIP Trunk** in Step 1, from the 'Template' drop-down list, select the interoperability configuration template.
4. From the 'Network Setup' drop-down list, select the physical network topology:
   - **Two ports: LAN and WAN:** The device connects to the network through two separate physical network links (interfaces). The first interface ("LAN") is connected to the Enterprise LAN (typically, a switch) and has a private IP address. The second interface ("WAN") is connected to the DMZ port of the Enterprise router and has a public (globally routable) IP address. Each link may be accompanied with a backup link for Ethernet link redundancy.
   - **One port: LAN:** The device connects to the Enterprise LAN (typically, a switch) through a single physical network link (interface). The interface ("LAN") has a private IP address. You must enable port forwarding on the Enterprise router to forward all VoIP traffic from the ITSP (located on the WAN) to the device. The exact port forwarding configuration is shown on the Conclusion page and consists of the device's address, SIP port (e.g. 5060) and a media port range (e.g. 6000-6999).
   - **One port: WAN:** The device connects to the DMZ port of the Enterprise router through a single physical network link (interface). The interface ("WAN") has a public (globally routable) IP address. You must enable port forwarding on the Enterprise router to forward all VoIP traffic from the device to the IP PBX (located on the LAN). The exact port forwarding configuration is shown on the Conclusion page and consists of the IP PBX address, SIP port (e.g. 5060) and a media port range (e.g. 6000-6999).
   - **One port: LAN only:** The device connects to the Enterprise LAN (typically, a switch) through a single physical network link (interface). All SIP entities (IP PBX and users) connect to the same LAN. Note that this option is applicable to all applications (see Step 1), except **SIP Trunk**.
5. Click **Next**; the System page appears (see System Page).

System Page

The System wizard page configures the device's basic system settings, including management interface protocol, date and time, and syslog.



➢ To configure system settings:

1. Configure the protocol for accessing the management interfaces:
   - 'Web Interface': Select the type of Web interface protocol (**HTTP** or **HTTPS**).
   - 'CLI Interface': Select the type of CLI protocol (**Disabled**, **Telnet**, **SSH**).
2. (Optional) Enable Syslog by selecting the 'Enable Syslog' check box, and then in the 'Syslog IP' field, configuring the Syslog server's IP address. Note that if you enable Syslog, the device generates verbose logs (debug level 5), which adversely affects performance. For more information on Syslog, see Configuring Syslog.
3. Configure the time and date:
   - 'Time Zone': Select the GMT time zone in which the device is located.
   - 'Primary NTP Server' and 'Secondary NTP Server': Configure the IP address (or hostname) of the primary (and optionally, secondary) NTP server in your network. The NTP server synchronizes the time of the device. This is mandatory when the IP PBX or ITSP uses a TLS connection, as correct time is required for certificate validation.

   For more information on configuring date and time, see Date and Time.
4. (Optional) Configure a local DNS table, which allows the device to resolve domain names (hostnames) using a locally defined address resolution table. This may be needed when your setup lacks a DNS server and the IP PBX or ITSP require the use of hostnames instead

of IP addresses. Select the 'Apply local DNS' check box, and then configure the following parameters:

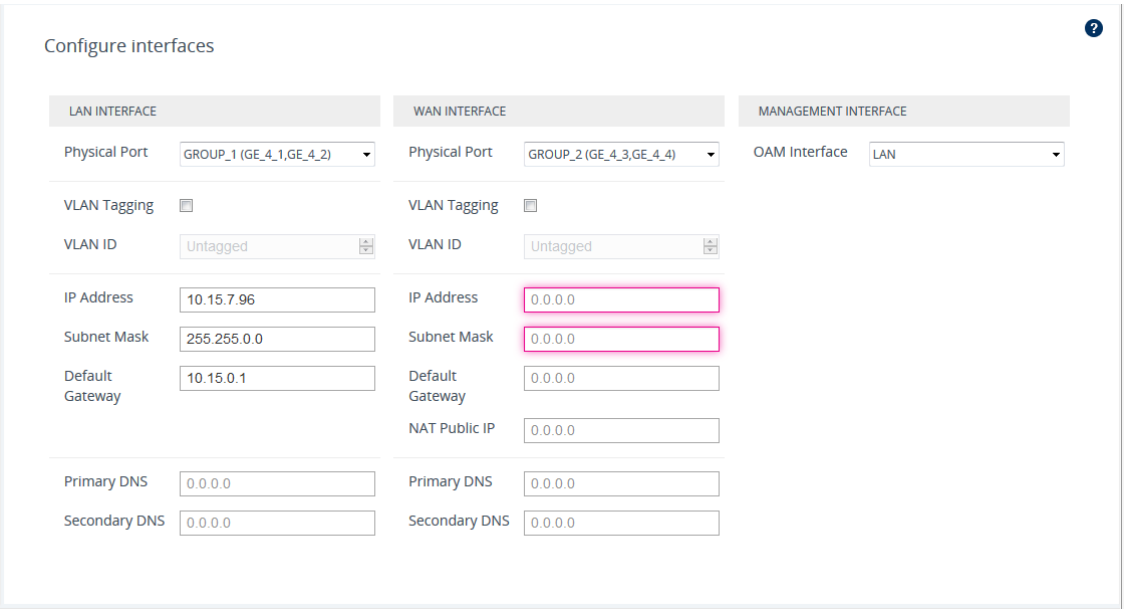
- 'Domain Name': Domain name to resolve into an IP address.
- 'First IP address': IP address of the domain name.
- 'Secondary IP address': Second IP address of the domain name (optional).

For more information on configuring the device's DNS table, see Configuring the Internal DNS Table.

5. Click **Next**; the Interfaces page appears (see Interfaces Page).

Interfaces Page

The Interfaces wizard page configures the device's LAN and WAN settings.



➢ To configure LAN and WAN interface settings:

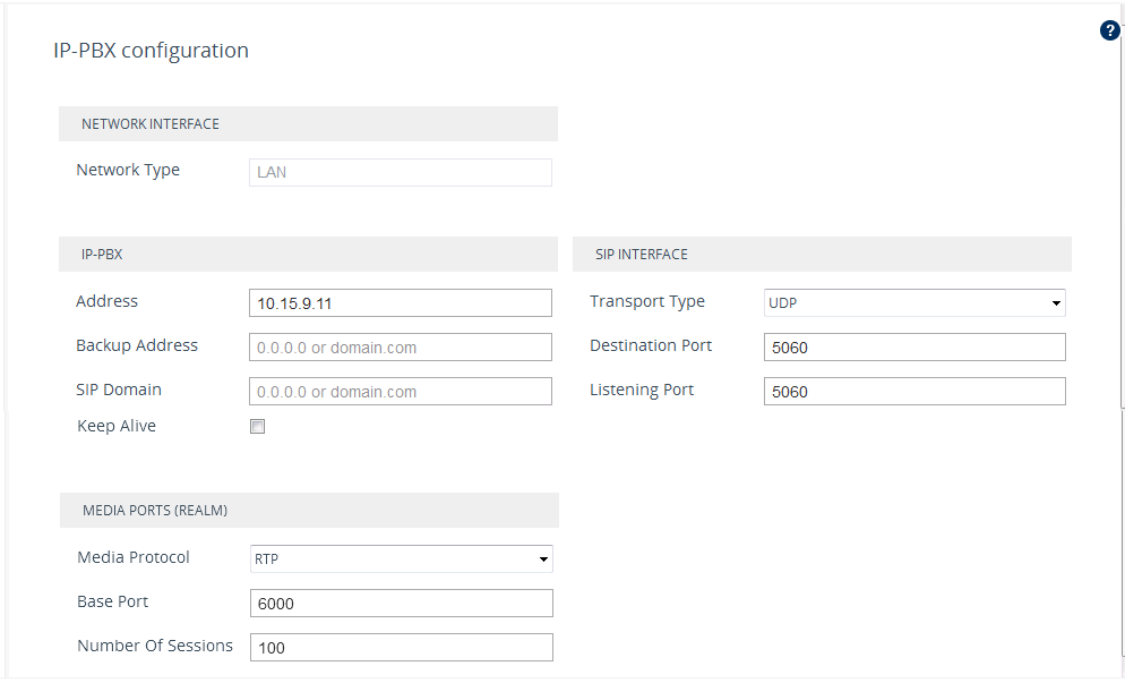
1. From the 'Physical Port' drop-down list, select the Ethernet Group containing the required physical Ethernet port for connecting the device to the 1) Enterprise LAN (typically, a switch) or 2) DMZ port of the Enterprise router for WAN. You may configure two physical ports for Ethernet link redundancy.
2. Select the **VLAN Tagging** check box, and then in the 'VLAN ID' field, configure the VLAN ID for the interface. For untagged traffic, clear the check box.
3. In the 'IP Address' field, configure the device's IP address on the interface.
   - For LAN interfaces: The IP address must be part of the Enterprise LAN and therefore, is typically a private IP address. The address is used for communicating with the IP PBX and/or users that reside on the LAN, as well as for management (OAMP) traffic.

- For WAN interfaces: The IP address must be a public (globally routable) address and is used for communicating with the ITSP and/or IP PBX that resides on the WAN. If the WAN is the only interface, it is also used for management (OAMP) traffic.

4. In the 'Subnet Mask' field, configure the subnet mask of the interface.
5. In the 'Default Gateway' field, configure the default gateway of the interface.
6. If the device is connected through an Enterprise router that performs NAT, then in the 'NAT Public IP' address, configure the public IP address (of the Enterprise router) used by the device to communicate with the ITSP (for the **SIP Trunk** application) or IP PBX (for the **Hosted IP-PBX** application).
7. In the 'Primary DNS Server' (and optionally, 'Secondary DNS Server') field, configure your primary (and optionally, secondary) DNS server in the network. This is mandatory if you use a hostname (FQDN) for ITSP (WAN only) and IP PBX addresses.
8. From the 'OAM Interface' drop-down list, select the device's interface for management traffic:
   - **LAN:** Management traffic is carried over the regular LAN interface, as defined above.
   - **WAN:** Management traffic is carried over the WAN interface, as defined above.
   - **Additional:** Configure a different interface for management traffic.
9. Click **Next**; the IP-PBX page appears (see IP-PBX Page).

IP-PBX Page

The IP-PBX wizard page configures the IP PBX settings.



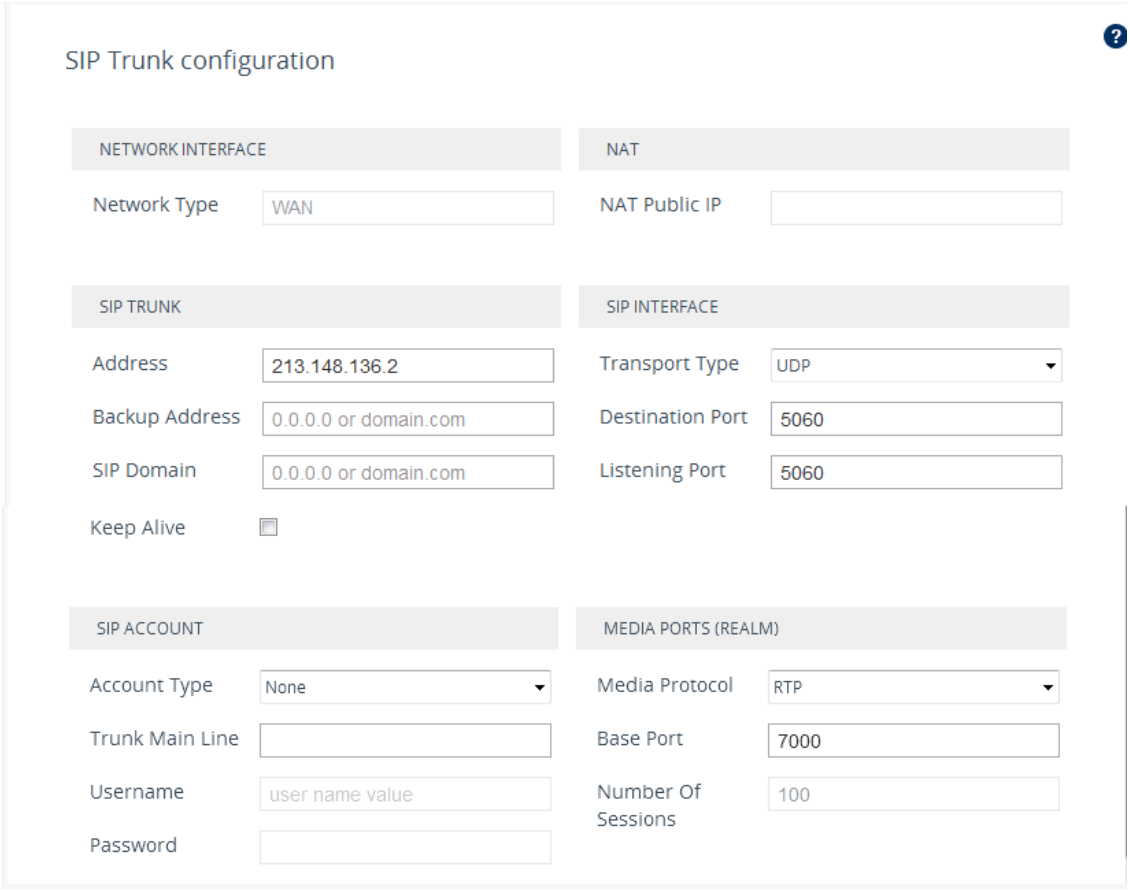
- The following fields are read-only:
  - ✓ 'Network Type': Displays the IP network interface for communicating with the IP PBX.
  - ✓ 'NAT Public IP': Displays the public IP address (of the Enterprise router) for communicating with the IP PBX. The field is applicable only when the device is connected to a router that performs NAT.
- Depending on the application type that you selected on the General Setup page (General Setup Page on page 929), the wizard may provide additional IP-PBX pages ("IP-PBX2" and "IP-PBX3") for configuring additional servers.

➢ **To configure IP PBX settings:**

1. Under the IP-PBX group, configure the following:
   - 'Address': Configure the IP address (or hostname) of the IP PBX. Note that for the **One port: WAN** network topology, when the device is assigned a public IP address, you must use the public IP address (of the Enterprise router) instead of the private address of the IP PBX, and configure the Enterprise router to forward VoIP traffic from the device to the IP PBX.
   - 'Backup Address': (Optional) Configure the backup IP address (or hostname) of the IP PBX.
   - 'SIP Domain': Configure the SIP domain name used for communication with the IP PBX. The domain name is used in the following SIP message headers:
     - Outbound calls: Request-URI and To headers
     - Inbound calls: From header
   - 'Keep Alive': Enable the periodic keep-alive check for multiple IP PBX addresses.
2. Under the Media Ports (Realm) group, configure the media protocol type and ports used by the device for communicating with the IP PBX:
   - 'Media Protocol': Configure the media protocol type (**RTP** or **SRTP**).
   - 'Base Port' Configure the first media port in the port range.
   - 'Number Of Sessions': Configure the number of required media sessions. For more information on media port ranges and number of sessions, see Configuring RTP Base UDP Port.
3. Under the SIP Interface group, configure SIP ports and transport type for communicating with the IP PBX:
   - 'Transport Type': Configure the SIP transport type.
   - 'Destination Port': Configure the SIP port used by the IP PBX.
   - 'Listening Port': Configure the SIP port used by the device when communicating with the IP PBX.
4. Click **Next**; the SIP Trunk page appears (SIP Trunk Page).

SIP Trunk Page

The SIP Trunk wizard page configures the SIP Trunk settings.



The following fields are read-only:

- 'Network Type': Displays the IP network interface for communicating with the SIP Trunk.
- 'NAT Public IP': Displays the public IP address (of the Enterprise router) for communicating with the SIP Trunk. The field is applicable only when the device is connected to a router that performs NAT. Note that the Enterprise router must be configured to "port forward" all VoIP traffic from the SIP Trunk (located on the WAN) to the device. The exact port forwarding configuration is displayed on the Conclusion page and consists of the device's address, SIP listening port (e.g. 5060) and a range of media ports defined below (e.g. 6000-6999).

➢ **To configure SIP Trunk settings:**

1. Under the SIP Trunk group, configure the following:
   - 'Address': Configures the IP address or hostname of the SIP Trunk.
   - 'Backup Address': (Optional) Configures the backup IP address or hostname of the SIP Trunk.

- 'SIP Domain': Configures the SIP domain name for communicating with the SIP Trunk. The domain name is used in the following SIP message headers:
  - Outbound calls: Request-URI and To headers
  - Inbound calls: From header
- 'Keep Alive': Enables the periodic keep-alive check of multiple SIP Trunk addresses.

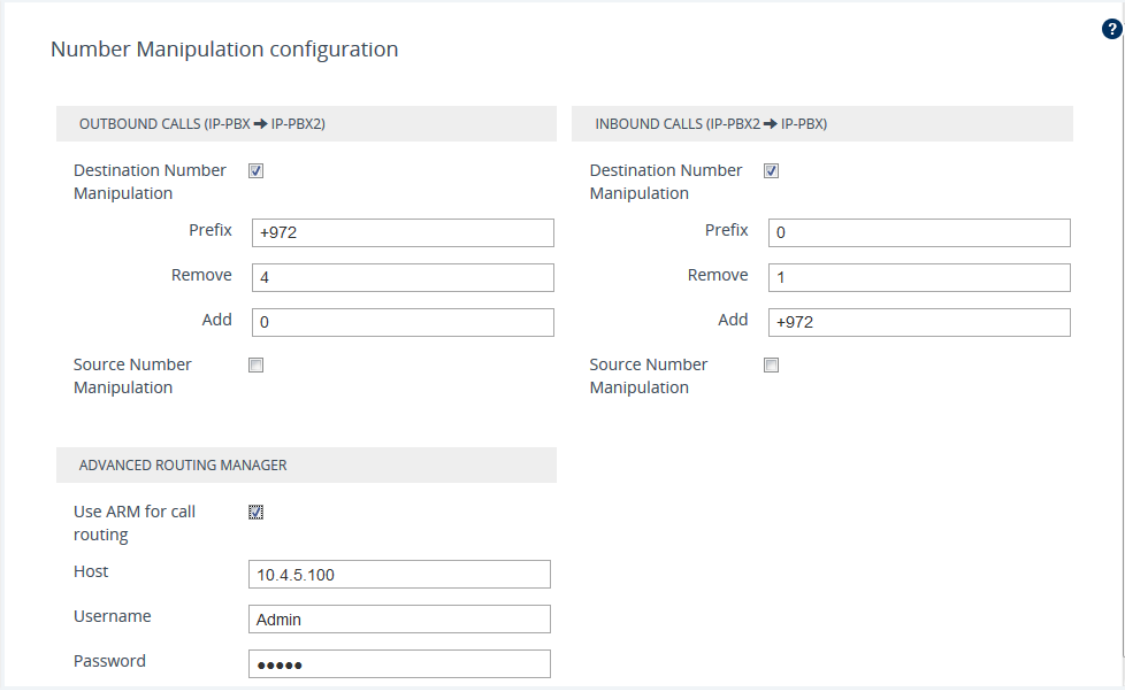
2. Under the SIP Interface group, configure the SIP ports and transport type for communicating with the SIP Trunk:
   - 'Transport Type: Configure the SIP transport type.
   - 'Destination Port: Configure the SIP port used by the SIP Trunk.
   - 'Listening Port: Configure the SIP port used by the device for communicating with the SIP Trunk. Note that for the **One port: WAN** network topology, the device must use different Listening Ports when communicating with the IP PBX and SIP Trunk.
3. Under the Media Ports (Realm) group, configure the media protocol type and ports used by the device for communicating with the IP PBX:
   - 'Media Protocol': Configure the media protocol type.
   - 'Base Port': Configure the first media port.
   - 'Number Of Sessions': Configure the number of required media sessions. For more information on media port ranges and number of sessions, see Configuring RTP Base UDP Port.
4. Under the SIP Account group, configure the device's registration with the SIP Trunk:
   - 'Account Type': Configure whether the device must perform registration or authentication with the SIP Trunk (**None**, **Registration** or **Authentication**).
   - 'Trunk Main Line': Configure the "leading number" assigned by the SIP Trunk. Many SIP Trunks use the same value for Trunk Main Line and Username parameters.
   - 'Username': Configure the SIP authentication username (as provided by the SIP Trunk provider).
   - 'Password': Configure the SIP authentication password (as provided by the SIP Trunk provider).

The password cannot be configured with wide characters.

5. Click **Next**; the Number Manipulation page appears (see Number Manipulation Page).

Number Manipulation Page

The Number Manipulation wizard page configures caller (source) and callee (destination) number manipulation for outbound and inbound calls, and allows you to use AudioCodes Routing Manager (ARM) to determine the routing.



➢ To configure number manipulation:

1. Configure number manipulation:
   a. Under the Outbound Calls and/or Inbound Calls groups, select the required manipulation check box:
      - Destination Number Manipulation: Manipulates the destination number.
      - Source Number Manipulation: Manipulates the source number.
   b. In the 'Prefix' field, configure the prefix (digits at the beginning of the number) to which you want to apply manipulation. If configured to "*" (asterisk), manipulation is applied to all numbers.
   c. In the 'Remove' field, configure the number of digits to be removed from the beginning of the number. If configured to "0", no digits are removed.
   d. In the 'Add' field, configure a new prefix to be added to the beginning of the number. If not configured, no prefix is added.

   The example below changes the number "+15033311432" to "03311432":
   - Prefix: "+1503"
   - Remove:"4"
   - Add: "0"
2. To enable routing by ARM, select the 'Use ARM for call routing' check box, and then configure the following fields:
   - 'Host': IP address or FQDN of the ARM server.
   - 'Username': Username for communication with ARM.

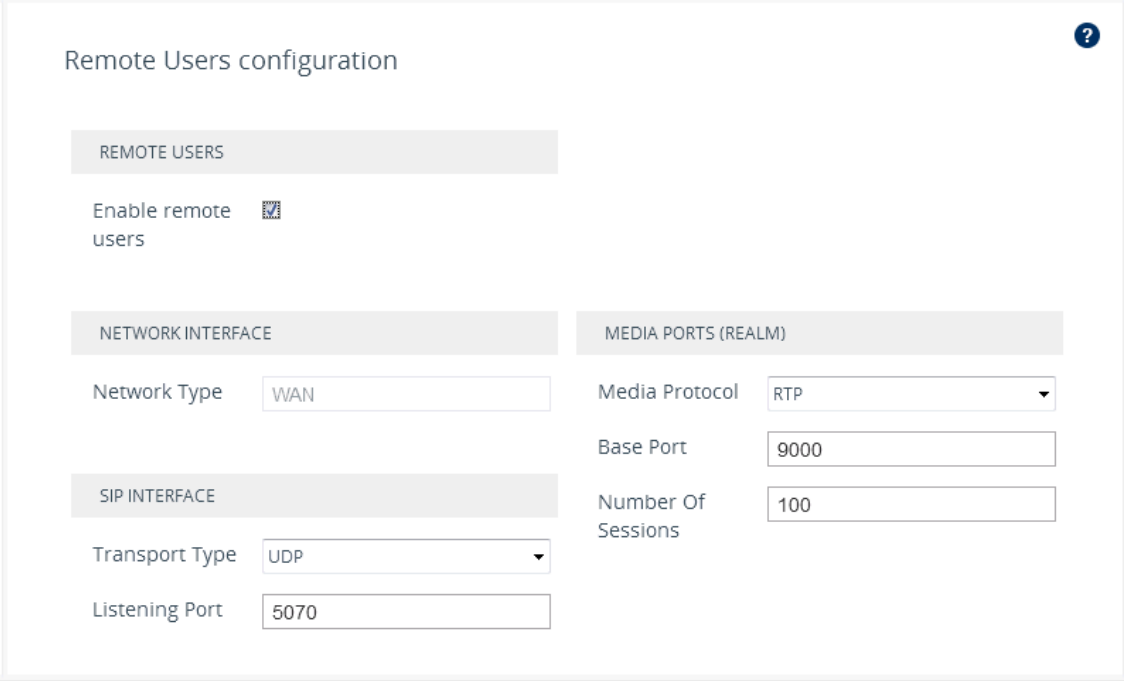
- 'Password': Password for communication with ARM.

3. Click **Next**; Remote Users (FEU) page appears (see Remote Users Page).

Remote Users or Users Page

The Remote Users or Users wizard page configures remote users settings.

- This page is applicable only to IP PBXs that support such configuration.
- The parameters displayed on the page depends on the application type and template that you selected on the General Setup wizard page (see General Setup Page on page 929).



➢ To configure remote users:

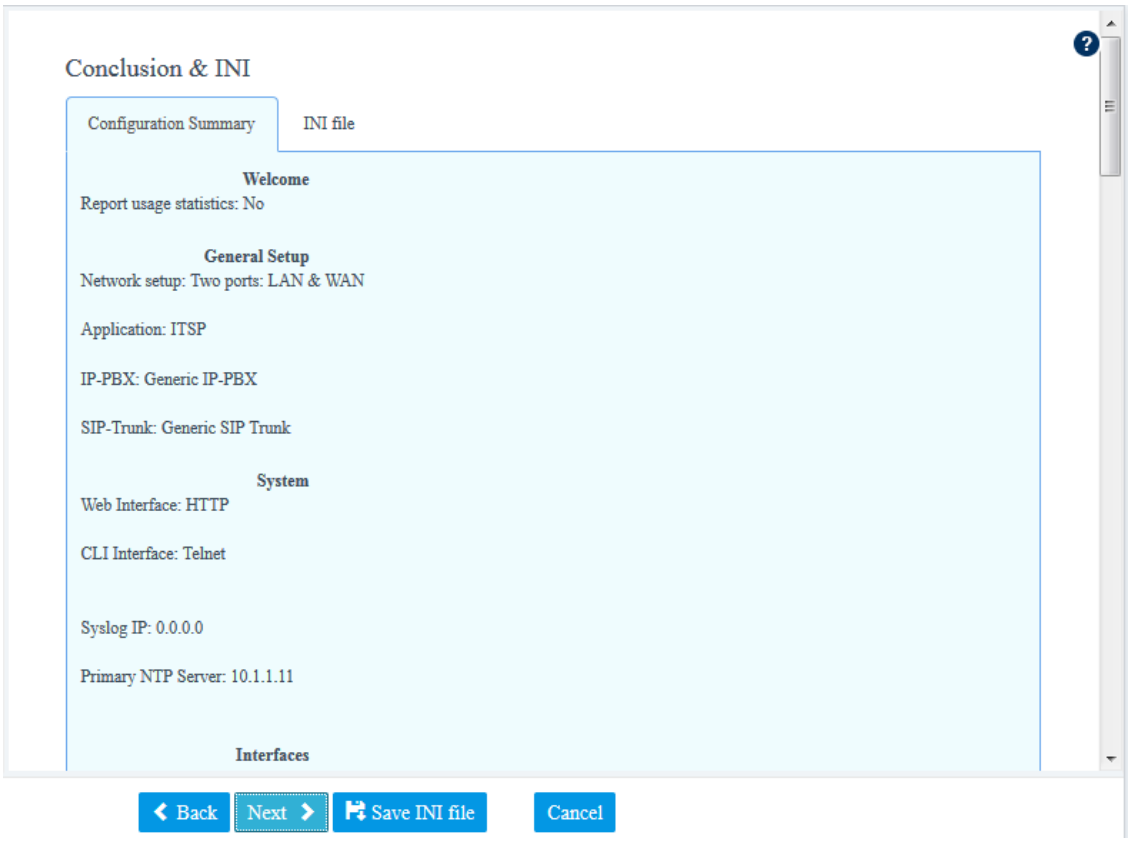
1. Select the 'Enable remote users' check box.
2. Under the SIP Interface group, configure the SIP interface for the remote users:
   - 'Transport Type': Configure the SIP transport type.
   - 'Listening Port': Configure the SIP port used by the device for communicating with remote users. For **One Port: LAN** and **One Port: WAN** network topologies, you must configure different listening ports for communication with the IP PBX and remote users.
3. Under the Media Ports (Realm) group, configure the media protocol type and ports used by the device for communicating with the remote users:
   - 'Media Protocol': Configure the media protocol type (RTP or SRTP).

- 'Base Port': Configure the first media port.
- 'Number Of Sessions': Configure the number of required media sessions. For more information on media port ranges and number of sessions, see Configuring RTP Base UDP Port.

4. Click **Next**; the Summary page appears (see Summary Page).

Summary Page

The Summary wizard page displays a summary of your configuration.

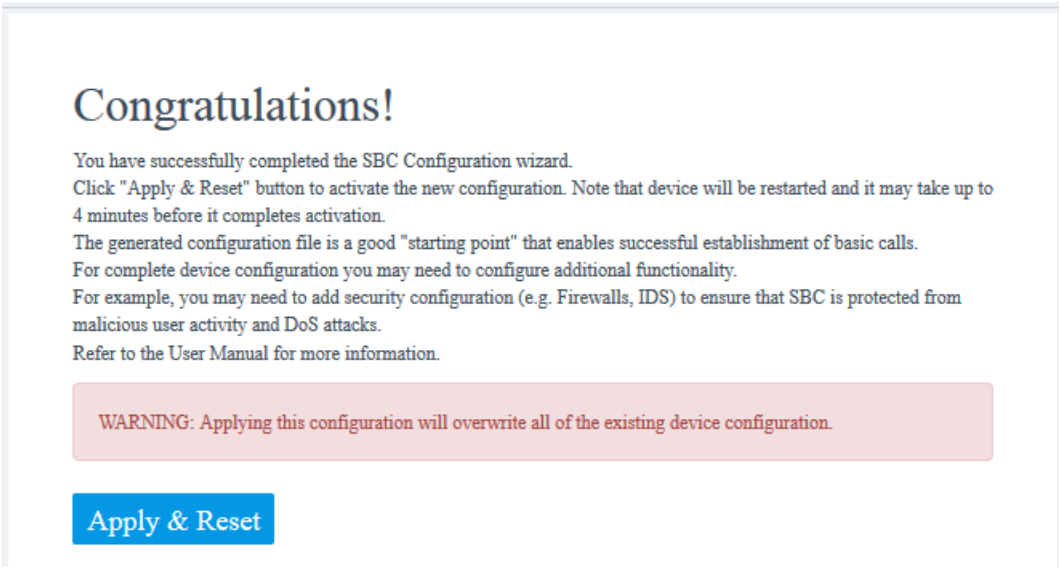


➢ To review your configuration:

1. Review the configuration:
   - To view the configuration in ini-file format, click the **INI File** tab.
   - To view the configuration in normal format, click the **Configuration Summary** tab.
2. To save the configuration as an ini file to a folder on your PC, click the **Save INI file** button. You can later load the file to the device (see Loading an ini File to the Device).
3. Click **Next**; the Congratulations page appears (see Congratulations Page).

Congratulations Page

The Congratulations wizard page is the last wizard page and allows you to complete configuration.



➢ **To complete the SBC Configuration Wizard:**

- Click **Apply & Reset** to apply configuration to the device or click **Save INI File** to save configuration as an ini file on your PC.

41 Restoring Factory Defaults

This section describes how to restore the device's configuration to factory defaults.

Restoring Factory Defaults through CLI

You can restore the device to factory defaults through CLI. You can restore all configuration to factory defaults or you can restore all configuration to factory defaults except the current network settings. Preserving network settings allows you to remotely connect to the device using its current OAMP IP address even after the device has been restored to default settings.

➢ **To restore factory defaults through CLI:**

1. Access the CLI:
   a. Connect the RS-232 serial port of the device to the communication port on your computer. For serial cabling, refer to the Hardware Installation Manual.
   b. Establish serial communication with the device using a serial communication program (such as HyperTerminalTM) with the following communication port settings:
      - Baud Rate: 115,200 bps
      - Data Bits: 8
      - Parity: None
      - Stop Bits: 1
      - Flow Control: None
2. At the CLI prompt, type the username (default is "Admin" - case sensitive), and then press Enter:

```
# Username: Admin
```

3. At the prompt, type the password (default is "Admin" - case sensitive), and then press Enter:

```
# Password: Admin
```

4. At the prompt, type the following, and then press Enter:

```
# enable
```

5. At the prompt, type the password again, and then press Enter:

```
# Password: Admin
```

6. At the prompt, type one of the following commands, and then press Enter:
   - To restore all configuration to factory defaults:

     ```
     # write factory
     ```

   - To restore configuration to factory defaults except current network settings:

     ```
     # write factory keep-network-and-users-configuration
     ```

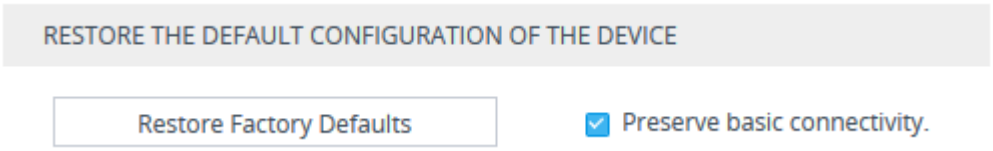
Restoring Factory Defaults through Web Interface

You can restore the device to factory defaults through the Web interface.

> When restoring the device to factory defaults, you can preserve basic IP network settings (configured in the IP Interfaces table - see Configuring IP Network Interfaces), as described below. This ensures that connectivity to the device (through the OAMP interface) is maintained after factory defaults have been applied.

➢ **To restore factory defaults through Web interface:**

1. Open the Configuration File page:
   - **Toolbar:** From the **Actions** drop-down menu, choose **Configuration File**.
   - **Navigation tree: Setup** menu > **Administration** tab > **Maintenance** folder > **Configuration File**.

2. To keep your current IP network settings (e.g., OAMP), select the **Preserve basic connectivity** check box. To overwrite all your IP network settings with the default IP network interface, clear this check box.
3. Click the **Restore Defaults** button; a message appears requesting you to confirm.
4. Click **OK** to confirm or **Cancel** to return to the page.
5. Once the device is restored to factory defaults, reset the device for the settings to take effect.

Restoring Defaults through ini File

You can restore the device to factory defaults as described below.

➢ **To restore the device to factory defaults:**

1. Create an **empty** text-based file and save it in a folder on your PC with the filename extension .ini.
2. Load the file to the device using the Configuration File page (see Configuration File).

The only settings that are not restored to default are the management (OAMP) LAN IP address and the Web interface's login username and password.

Part VIII

Status, Performance Monitoring and Reporting

42 System Status

This section describes how to view system status.

Viewing Device Information

You can view hardware and software information about the device on the Device Information page.

➢ **To view device information:**

- Open the Device Information page (**Monitor** menu > **Monitor** tab > **Summary** folder > **Device Information**).

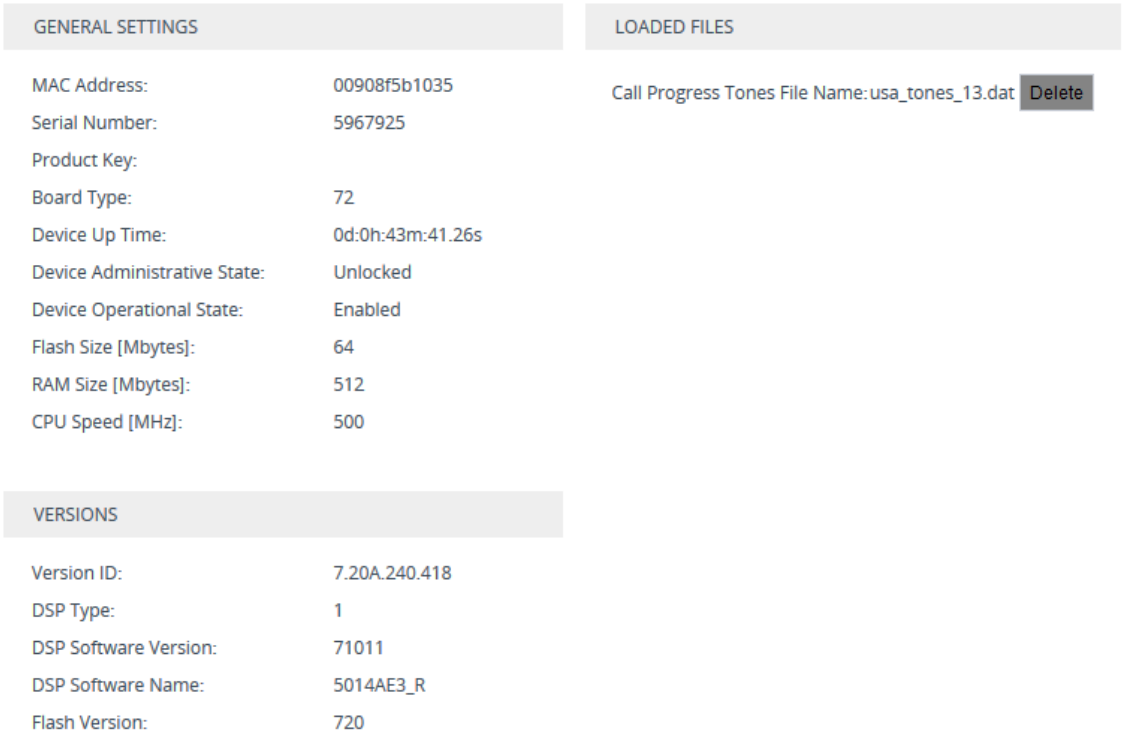


Table 42-1: Device Information Description

Parameter	Description
General Settings	
MAC Address	Media access control (MAC) address.
Serial Number	Serial number of the CPU. This serial number also appears on the product label that is affixed to the chassis.
Product Key	Product Key, which identifies the specific device purchase (and used for communication with AudioCodes, for example, for support and

Parameter	Description
	software upgrades). The Product Key also appears on the product label that is affixed to the chassis. For more information, see Viewing the Device's Product Key.
Board Type	Numerical identification of the product (device).
Device Up Time	Duration that the device has been up and running since the last reset (uptime). The duration is displayed in the following format: *dd:hh:mm:ss.ss*. For example, "1d:21h:40m:21s:75.22" means that the device has been running for one day and 21 hours, 40 minutes and 21.22 seconds.
Device Administrative State	Administrative status ("Unlocked" or "Locked"), as performed in Locking and Unlocking the Device.
Device Operational State	Operational status: ■ "Disabled" ■ "Enabled" ■ "Error" ■ "Unknown"
Flash Size [Mbytes]	Size of the non-volatile storage memory (flash), measured in megabytes.
RAM Size [Mbytes]	Size of the random access memory (RAM), measured in megabytes.
CPU Speed [MHz]	Clock speed of the CPU, measured in megahertz (MHz).
Versions	
Version ID	Software version number.
DSP Type	Type of DSP.
DSP Software Version	DSP software version.
DSP Software Name	DSP software name.
Flash Version	Flash memory version number.

Parameter	Description
Loaded Files: Displays installed Auxiliary files. You can also delete a file, by clicking the corresponding **Delete** button, as described in Deleting Auxiliary Files.	

Viewing Device Status on Monitor Page

The Web interface's Monitor page provides basic status and information on the device. The page is useful in that it allows you to easily obtain an overview of the device's operating status at a glance.

➢ **To view device status and information on the Monitor home page:**

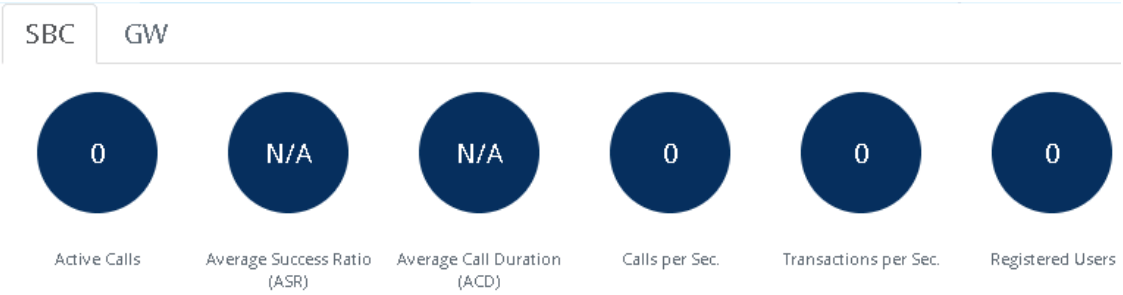
- On the Menu bar, click **Monitor** or if you are already in the Monitor menu's Navigation tree, click **Monitor**.

The Monitor page displays the following groups of information:

- **Device Information:**
  - **Address:** IP address of the device's OAMP interface
  - **Firmware:** Software version currently running on the device
  - **Type:** Name of the device
  - **HA Status:** High-Availability (HA) status of the device, if configured for HA. For more information, see Viewing HA Status on Monitor Web Page.
  - **S/N:** Serial number of the device.
- **SBC Call Statistics:**
  - **Active Calls:** Total number of SBC calls. The corresponding SNMP performance monitoring MIB is PM_gwINVITEDialogs.
  - **Average Success Rate (ASR):** Number of successfully answered calls out of the total number of attempted calls. The corresponding SNMP performance monitoring MIB is PM_gwSBCASR.
  - **Average Call Duration (ACD):** Average call duration in seconds of established calls. The value is refreshed every 15 minutes and therefore, this value reflects the average duration of all established calls made within a 15 minute period. The corresponding SNMP performance monitoring MIB is PM_gwSBCACD.
  - **Calls per Sec:** Total number of new calls per second (CPS).
  - **Transactions per Sec:** Total number of new SIP transactions per second (out-of-dialog transactions such as INVITE and REGISTER, or in-dialog transactions such as UPDATE and BYE). The corresponding SNMP performance monitoring MIB is PM_gwActiveSIPTransacionsPerSecond.

- **Registered Users:** Number of users registered with the device. The corresponding SNMP performance monitoring MIB is acPMSBCRegisteredUsersTable.

Figure 42-1: SBC



- **Graphical Display of device:** Shows color-coded status icons, as shown in the figure below and described in the subsequent table:

Figure 42-2: Graphical Display of Device (Example) on Monitor Page - Mediant 4000 SBC

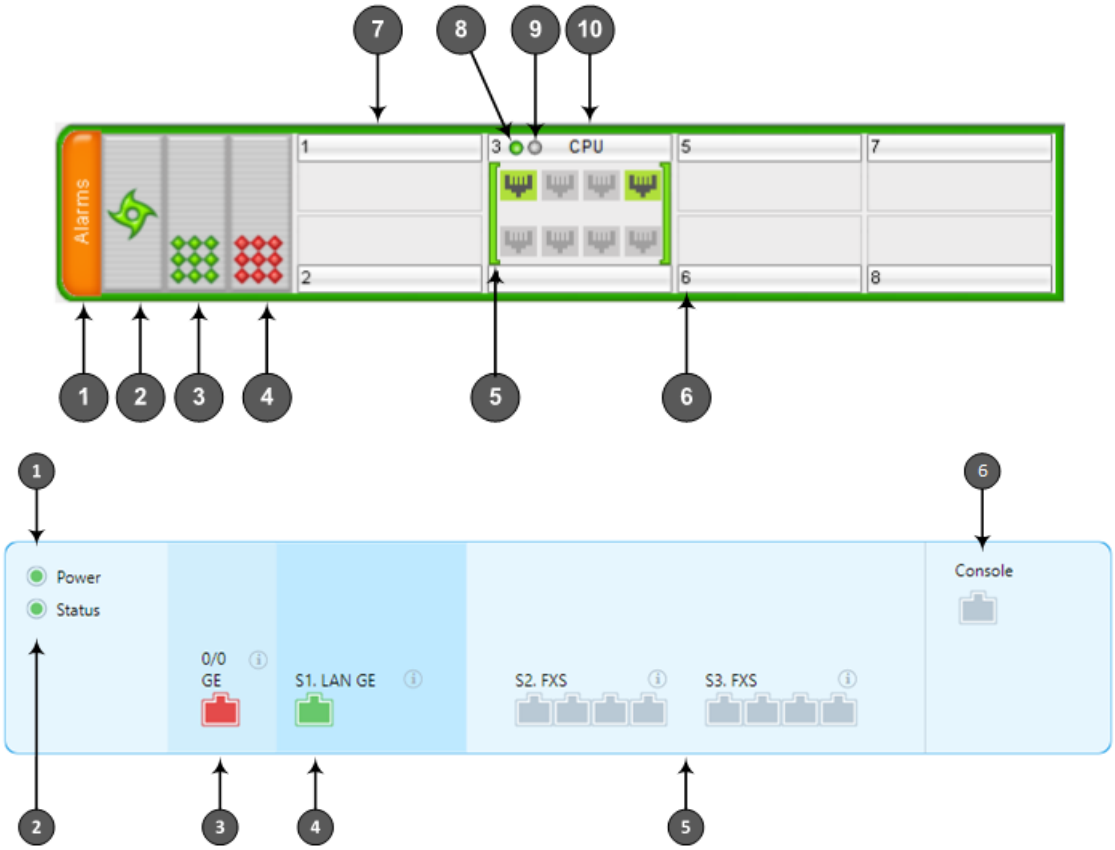
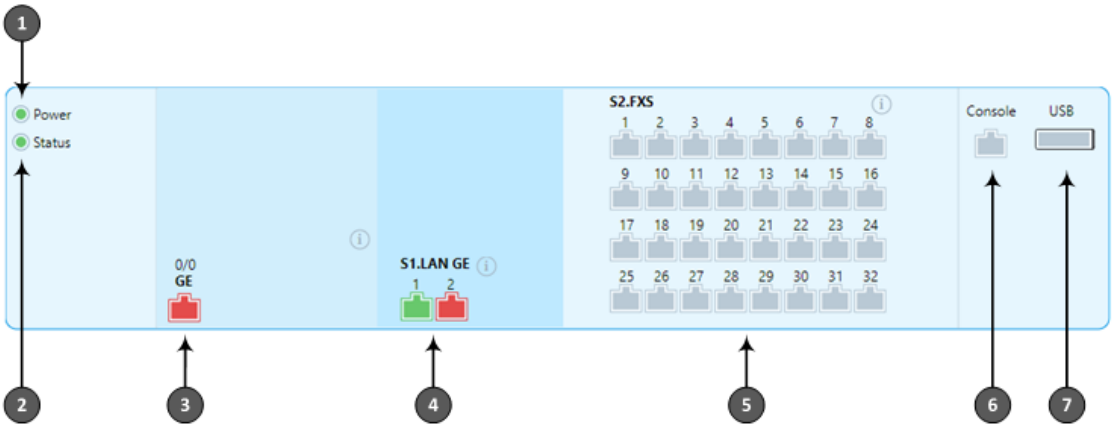


Figure 42-3: Graphical Display of Device (Example) on Monitor Page - MP-532



- The figure above is used only as an example as the graphical display of your device in the Web interface reflects your specific ordered hardware configuration.
- For a description of the Monitor page when the device is in High Availability (HA) mode, see HA Status Display on Monitor Web Page.

Table 42-2: Description of Graphical Display of Device on Monitor Page

Item #1	Description
1	**Alarms:** Displays the highest severity of an active alarm raised (if any) by the device: ■ Green = no alarms ■ Red = Critical alarm ■ Orange = Major alarm ■ Yellow = Minor alarm To view active alarms, click **Alarms** to open the Active Alarms page (see Viewing Active Alarms).
2	Fan tray unit status icon: ■ (green): Normal operation of Fan tray. ■ (red): Fan tray failure. To view detailed information of the device's hardware components, click these icons to open the Components Status page (see Viewing Hardware Components Status.
3	Power Supply Module 2 status icon:

Item #1	Description
	■  (green): Normal operation of Power Supply module. ■  (red): Failure in Power Supply module or no Power Supply module is installed. To view detailed information of the device's hardware components, click these icons to open the Components Status page (see Viewing Hardware Components Status).
4	Power Supply module 1 status icon. For more information, see the description above.
5	Module status icon: ■  (green): Module has been inserted or is correctly configured ■  (gray): Module was removed and "Reserved" is displayed ■  (red): Module failure and "Failure" is displayed
6	Chassis slot number.
7	Optional, Media Processing Module (MPM). **Note:** This is a customer-ordered module.
8	OK LED icon: ■  (red): Device undergoing startup. ■  (green): Device startup has completed successfully.
9	Active LED icon: ■  (gray): No High Availability (HA). This appears if the HA feature is not included in the installed License Key, HA is not configured, or upon HA initialization failure. ■  (green): Applicable only to the graphical representation of the Active device in HA mode and means one of the following: ✓ The Active device is operating in Standalone mode (i.e., Redundant device is missing and HA is unavailable) ✓ The Active device is operating in HA mode (i.e., Redundant device exists and HA is available)

Item #1	Description
	■ (yellow): Applicable only to the graphical representation of the Redundant device in HA mode and means one of the following: ✓ The Redundant device is still synchronizing with the Active device (i.e., HA is unavailable until HA synchronization finishes). ✓ The Redundant device is operating successfully in HA mode
10	SBC module, providing the Ethernet port status icons: ■  (green): Ethernet link is working ■  (gray): Ethernet link is not connected To view detailed Ethernet port information, click these icons to open the Ethernet Port Information page (see Viewing Ethernet Port Information).

43 Reporting DSP Utilization through SNMP MIB

You can obtain information on the percentage of DSP resources utilized by the device, through the SNMP MIB table, acPMDSPUsage. You can also configure low and high DSP utilization thresholds for this MIB, that if crossed, the SNMP trap event, acPerformanceMonitoringThresholdCrossing is sent by the device. For more information on this MIB, refer to the *SNMP Reference Guide*.

44 Viewing Carrier-Grade Alarms

This section describes how to view SNMP alarms raised by the device.

Viewing Active Alarms

You can view current (active) alarms in the Web interface that have been raised by the device. If an alarm is cleared, it is moved into the History Alarms table (see Viewing History Alarms). The alarms are displayed from newest to oldest. In other words, the most recently raised alarm is shown first in the list. The table is automatically refreshed every 60 seconds.

If the device is in High Availability (HA) mode, the table also displays alarms raised by the Redundant device. These alarms are shown in the 'Source' field with the alarm source varbind prefix, "Redundant" (e.g., "Redundant#1/EthernetLink#7"). Alarms raised by the Active device are shown without this source varbind prefix (e.g., Board#1/EthernetLink#7).

- The alarms in the table are deleted upon a device reset.
- If the device is in HA mode, the Redundant device's alarms in the table are deleted upon an HA switchover or if the device changes from HA to a Standalone device.
- To configure the maximum number of active alarms that can be displayed in the table, see the ini file parameter, ActiveAlarmTableMaxSize.
- The alarm bell icon, located on the top-right of the Web interface's window, displays the number of currently active alarms raised by the device and the highest severity (color coded - see below) of these alarms. If the device is in HA mode, the icon displays the number of currently active alarms raised by both Active and Redundant devices as well as the highest severity of all these active alarms.
- For more information on SNMP alarms, refer to the *SNMP Reference Guide* document.

➢ **To view active alarms:**

1. Open the Active Alarms table:
   - Navigation tree: **Monitor** menu > **Monitor** tab > **Summary** folder > **Active Alarms.**
   - Monitor home page: Click the "Alarms" area on the graphical display of the device (see Viewing Device Status on Monitor Page).
   - Alarm bell 

icon (located in the top-right area of the Web interface)

SEQUENTIAL #	SEVERITY	SOURCE	DESCRIPTION	TIME
13	Minor	Board#1/HTTPProxyService#0	HTTP Proxy Upstream Host 10.1.1.1.1:45(Host #0 in Upstream Group #0) is OF	04/01/2010, 00:12:23
12	Major	Board#1/HTTPProxyService#0	NGINX Configuration file is not valid. Nginx configuration file is Not Valid	04/01/2010, 00:12:11
5	Major	Board#1/ProxyConnection#1	Proxy Set Alarm Proxy Set 1 (IP-PBX): Proxy lost. looking for another proxy	01/01/2010, 00:00:53
4	Minor	Board#1/EthernetLink#4	Ethernet link alarm. LAN port number 4 is down.	01/01/2010, 00:00:51
3	Minor	Board#1/EthernetLink#3	Ethernet link alarm. LAN port number 3 is down.	01/01/2010, 00:00:51
2	Minor	Board#1/EthernetLink#2	Ethernet link alarm. LAN port number 2 is down.	01/01/2010, 00:00:51
1	Major	Board#1/ProxyConnection#2	Proxy Set Alarm Proxy Set 2 (ITSP-2): Proxy lost. looking for another proxy	01/01/2010, 00:00:50

Table 44-1: Active Alarms Table Description

Field	Description
Sequential #	The number of the alarm. Alarms are numbered sequentially as they are raised by the device. The numbering resets to 1 after a device reset (i.e., the first alarm raised after a reset is assigned the number #1).
Severity	Severity level of the alarm: ■ Critical (red) ■ Major (orange) ■ Minor (yellow)
Source	Component of the device from which the alarm was raised.
Description	Brief description of the alarm.
Date	Date (DD/MM/YYYY) and time (HH:MM:SS) the alarm was raised.

Viewing History Alarms

You can view all SNMP alarms, in the Web interface's Alarms History table, that have been raised (active alarms) as well as cleared (resolved). One of the benefits of this is that you can view alarms that may have been raised and then cleared on a continuous basis. For example, such an alarm may be raised due to an Ethernet cable that is not securely attached to the device's Ethernet port, causing the Ethernet link to be sometimes up and sometimes down. This alarm would not be listed in the Active Alarms table due to it being cleared.

The alarms in the table are displayed from newest to oldest. In other words, the most recently raised alarm is shown first in the list. The table displays both the cleared alarm and the alarm for which it was cleared adjacent to one another, as shown in the figure below for alarms #8 and #9.

To configure the maximum number of alarms that can be displayed in the table, use the AlarmHistoryTableMaxSize ini file parameter. If the maximum is reached and a new alarm is added to the table, the oldest alarm is removed from the table to accommodate the new alarm.

If the device is in High Availability (HA) mode, the table also displays alarms raised by the Redundant device. These alarms are shown in the 'Source' field with the alarm source varbind prefix, "Redundant" (e.g., "Redundant#1/EthernetLink#7"). Alarms raised by the Active device are shown without this source varbind prefix (e.g., Board#1/EthernetLink#7).

- The alarms in the table are deleted upon a device reset.
- If the device is in HA mode, the Redundant device's alarms in the table are deleted upon an HA switchover or if the device changes from HA to a Standalone device.
- For more information on SNMP alarms, refer to the *SNMP Reference Guide* document.

➢ To view history alarms:

- Open the Alarms History table (**Monitor** menu > **Monitor** tab > **Summary** folder > **Alarms History**).

SEQUENTIAL #	SEVERITY	SOURCE	DESCRIPTION	TIME
13	Minor	Board#1/HTTPProxyService#0	HTTP Proxy Upstream Host 10.1.1.1.1:45(Host #0 in Upstream Group #0) is OF	04/01/2010, 00:12:23
12	Major	Board#1/HTTPProxyService#0	NGINX Configuration file is not valid. Nginx configuration file is Not Valid	04/01/2010, 00:12:11
11	Cleared	Board#1/HTTPProxyService#0	HTTP Proxy Upstream Host 10.1.1.1:77(Host #0 in Upstream Group #0) is ONL	02/01/2010, 23:45:15
10	Cleared	Board#1/HTTPProxyService#0	Nginx configuration file is Valid	02/01/2010, 23:45:14
9	Major	Board#1/HTTPProxyService#0	NGINX Configuration file is not valid. Nginx configuration file is Not Valid	02/01/2010, 23:26:57
8	Cleared	Board#1/HTTPProxyService#0	Nginx configuration file is Valid	02/01/2010, 02:59:36
7	Major	Board#1/HTTPProxyService#0	NGINX Configuration file is not valid. Nginx configuration file is Not Valid	02/01/2010, 02:46:21
6	Minor	Board#1/HTTPProxyService#0	HTTP Proxy Upstream Host 10.1.1.1:77(Host #0 in Upstream Group #0) is OFF	02/01/2010, 00:46:24
5	Major	Board#1/ProxyConnection#1	Proxy Set Alarm Proxy Set 1 (IP-PBX): Proxy lost. looking for another proxy	01/01/2010, 00:00:53
4	Minor	Board#1/EthernetLink#4	Ethernet link alarm. LAN port number 4 is down.	01/01/2010, 00:00:51

Table 44-2: Alarms History Table Description

Field	Description
Sequential #	The number of the alarm. The alarms are numbered sequentially as they are raised by the device. The numbering resets to 1 immediately after a device reset (i.e., the first alarm raised after a reset is assigned the number #1).
Severity	Severity level of the alarm: ■ Critical (red) ■ Major (orange) ■ Minor (yellow) ■ Cleared (green)
Source	Component of the device from which the alarm was raised.
Description	Brief description of the alarm.
Date	Date (DD/MM/YYYY) and time (HH:MM:SS) the alarm was raised.

➢ To delete all the alarms in the table:

1. Click the **Delete History Table** button; a confirmation message box appears.

2. Click **OK** to confirm.

45 Viewing Management User Activity Logs

If you have enabled the reporting of management user activities performed in the device's management interfaces (see Configuring Reporting of Management User Activities), you can view the logged activities in the Web interface, as described in the procedure below.

➢ **To view management user activity logs:**

- Open the Activity Log table (**Monitor** menu > **Monitor** tab > **Summary** folder > **Activity Log**).

ID	TIME	DESCRIPTION	USER	INTERFACE	CLIENT
7	11/29/2010, 11:44:45	CLI: 'enable'	Admin	Telnet	10.13.2.3
6	11/29/2010, 11:44:43	User login succeeded	Admin	Telnet	10.13.2.3
5	11/29/2010, 11:40:38	System configuration has be	Admin	WEB	10.13.2.3
4	11/29/2010, 11:40:27	WEB: Successful login at 10	Admin	WEB	10.13.2.3
3	11/29/2010, 11:40:24	WEB: User logout	Admin	WEB	10.13.2.3
2	11/29/2010, 11:40:16	SRTP Tunneling Authenticati	Admin	WEB	10.13.2.3
1	11/29/2010, 11:39:58	System configuration has be	Admin	WEB	10.13.2.3

Table 45-1: Activity Log Table Description

Parameter	Description
Time	Date (mm/dd/yyyy) and time (hh:mm:ss) that the activity was performed.
Description	Description of the activity.
User	Username of the user account that performed the activity.
Interface	Protocol used for connecting to the management interface (e.g., Telnet, SSH, Web, or HTTP).
Client	IP address of the client PC from where the user accessed the management interface.

46 Viewing Performance Monitoring

This section describes how to view performance monitoring in the device's Web interface.

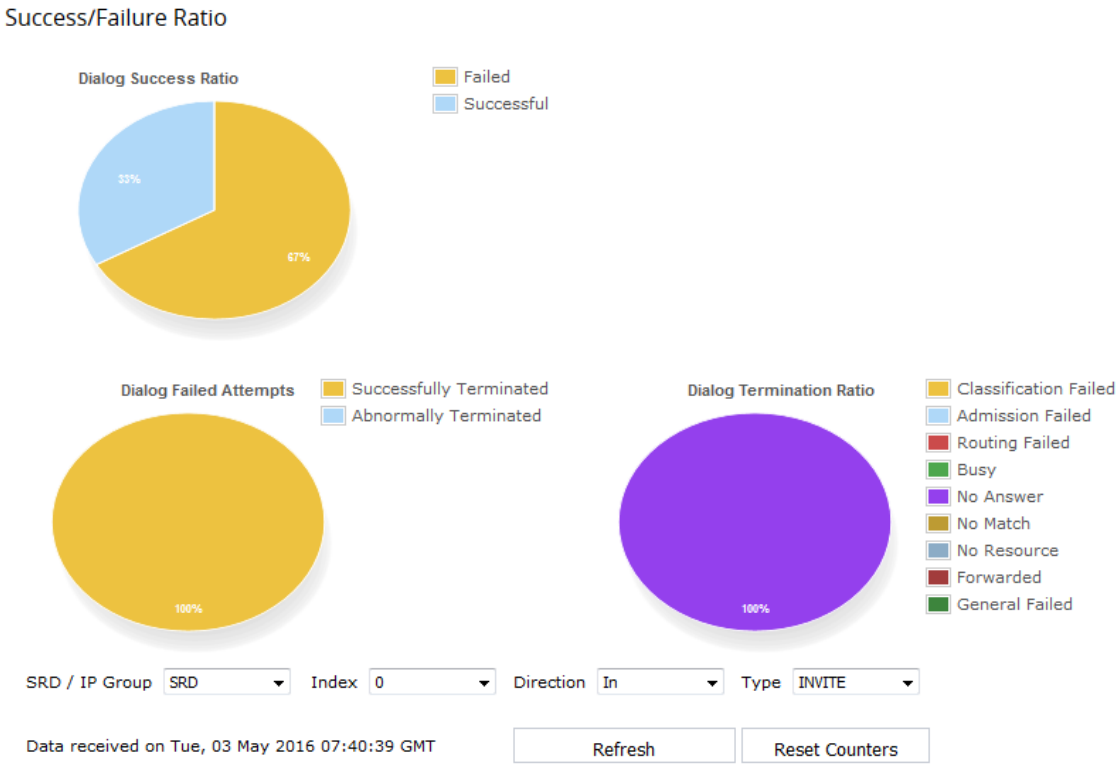
Viewing Call Success and Failure Ratio

You can view success and failure ratio of SIP dialogs in the Web interface's Success/Failure Ratio page. You can filter the display by SRD or IP Group, and by call direction and type of SIP dialog (e.g., INVITEs only). The information is displayed in the following pie charts:

- **Dialog Success Ratio:** Displays the SIP call and subscribe (SUBSCRIBE) dialog success-failed ratio.
- **Dialog Failed Attempts:** Displays failed SIP dialog attempts. This includes the number of calls and subscribes which were successfully and abnormally terminated.
- **Dialog Termination Ratio:** Displays SIP dialog termination by reason (e.g., due to no answer).

➢ **To view success and failed call ratio:**

1. Open the Success/Failure Ratio page (**Monitor** menu > **Monitor** tab > **Performance Monitoring** folder > **Success / Failure Ratio**).



2. From the 'SRD / IP Group' drop-down list, select whether you want to view statistic for an SRD or IP Group.
3. From the 'Index' drop-down list, select the SRD or IP Group index.

4. From the 'Direction' drop-down list, select the call direction:
   - **In:** incoming calls
   - **Out:** outgoing calls
   - **Both:** incoming and outgoing calls
5. From the 'Type' drop-down list, select the SIP message type:
   - **INVITE:** INVITE
   - **SUBSCRIBE:** SUBSCRIBE
   - **Other:** all SIP messages

If there is no data for the charts, the chart appears gray and "No Data" is displayed to the right of the chart.

➢ **To refresh the charts:**

- Click **Refresh**.

➢ **To reset the counters:**

- Click **Reset Counters**.

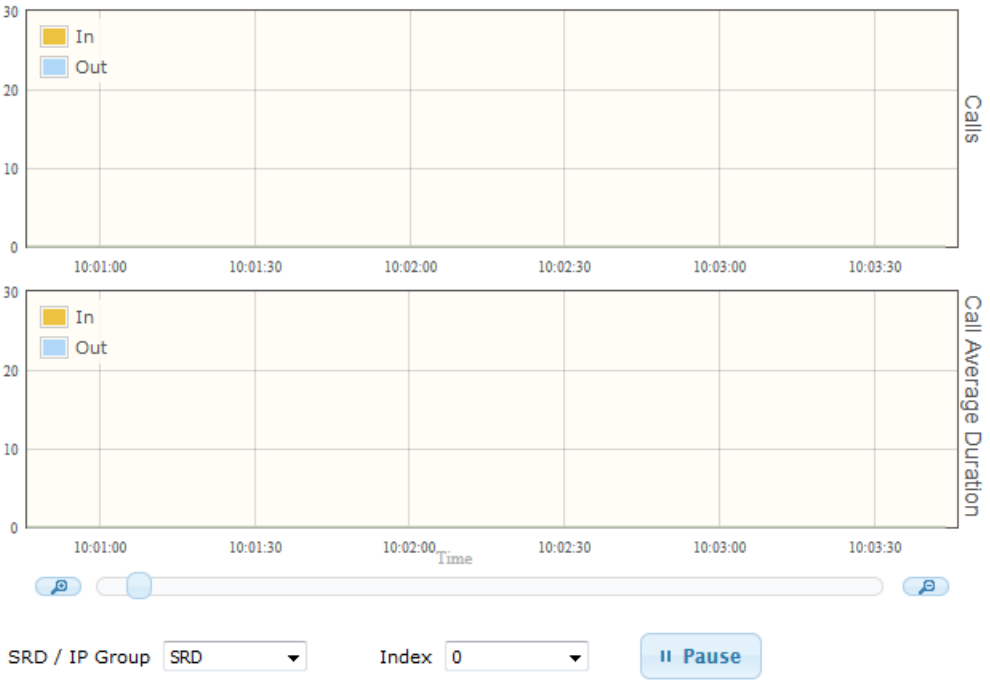
Viewing Average Call Duration

You can view the number of established calls over a 15-minute interval and the average call duration (ACD) in the Web interface's Average Call Duration page. You can filter display by a specific SRD or IP Group. The page displays the following two graphs:

- **Upper graph:** Displays the number of established calls (INVITEs) in a 15-minute interval. The x-axis indicates the time (hh:mm:ss) of the call and the y-axis the number of calls. The graph is refreshed every 15 minutes.
- **Lower graph:** Displays the ACD. The x-axis indicates the time (hh:mm:ss) and the y-axis the average call duration. The ACD is refreshed every 15 minutes and therefore, this value reflects the average duration of all established calls made within a 15-minute interval.

➢ **To view number of active calls and average call duration:**

1. Open the Average Call Duration page (**Monitor** menu > **Monitor** tab > **Performance Monitoring** folder > **Average Call Duration**).



2. From the 'SRD / IP Group' drop-down list, select the configuration entity (SRD or IP Group).
3. From the 'Index' drop-down list, select the specific SRD or IP Group index.

Use the **Zoom In** button to increase the displayed time resolution or the **Zoom Out** button to decrease it. Instead of using these buttons, you can use the slide ruler. As you increase the resolution, more data is displayed on the graph. The minimum resolution is about 30 seconds; the maximum resolution is about an hour.

To pause the graph, click the **Pause** button; click **Play** to resume.

Configuring Performance Profiles

The Performance Profile table lets you configure up to 2628 Performance Profile rules. A Performance Profile rule defines thresholds of performance monitoring call metrics for Major and Minor severity alarms. If the threshold is crossed, the device raises the corresponding severity alarm. You can configure a Performance Profile rule for all calls (*globally*), or per SRD or IP Group.

You can configure the alarm thresholds for the following call metrics:

- **Answer Success Ratio or ASR (also known as Answer Seizure Ratio):** The number (in percentage) of answered calls (i.e. number of seizures resulting in an answer signal) out of the total number of attempted calls (seizures). The metric is calculated for the outgoing call leg. The metric includes the following SNMP performance monitoring MIBs:
  - PM_gwSBCASR: ASR for all (global) entities (i.e., all IP Groups and SRDs)
  - PM_gwSBCIPGroupASR: ASR per IP Group
  - PM_gwSBCSRDASR: ASR per SRD

If the configured ASR minor or major thresholds are crossed, the device raises the SNMP alarm, acASRThresholdAlarm (OID 1.3.6.1.4.1.5003.9.10.1.21.2.0.111).

To view ASR in the Web interface, see Viewing Call Success and Failure Ratio.

- **Network Effectiveness Ratio (NER):** The number (in percentage) of successfully connected calls out of the total number of attempted calls (seizures). The metric measures the ability of the network to deliver a call to the called terminal. In addition to answered calls, the following SIP response codes are regarded as successfully connected calls: 408 (Request Timeout), 480 (Temporarily Unavailable), and 486 (Busy Here). The metric is calculated for the outgoing call leg. The metric includes the following SNMP performance monitoring MIBs:
  - PM_gwSBCNER: NER for all (global) entities (i.e., all IP Groups and SRDs)
  - PM_gwSBCIPGroupNER: NER per IP Group
  - PM_gwSBCSRDNER: NER per SRD

  If the configured NER minor or major thresholds are crossed, the device raises the SNMP alarm, AcNERThresholdAlarm (OID 1.3.6.1.4.1.5003.9.10.1.21.2.0.113).
- **Average Call Duration (ACD):** The ACD plus the session disconnect time (SDD) is the duration from when the SIP 200 OK is received to when the SIP Bye message is sent. The metric is calculated for both incoming and outgoing call legs. The metric includes the following SNMP performance monitoring MIBs:
  - PM_gwSBCACD: ACD for all (global) entities (i.e., all IP Groups and SRDs)
  - PM_gwSBCIPGroupACD: ACD per IP Group
  - PM_gwSBCSRDACD: ACD per SRD

  If the configured ACD minor or major thresholds are crossed, the device raises the SNMP alarm, acACDThresholdAlarm (OID 1.3.6.1.4.1.5003.9.10.1.21.2.0.112).

  To view ACD in the Web interface, see Viewing Average Call Duration.

At any given time during a call, a voice metric can be in one of the following color-coded quality states (as displayed in OVOC):

- **Green:** Indicates good call quality
- **Yellow:** Indicates fair call quality
- **Red:** Indicates poor call quality

When the threshold of a voice metric is crossed, the device changes the alarm severity and corresponding color-coded quality state of the call:

- **Minor Threshold (Yellow):** Lower threshold that indicates changes from Green or Red to Yellow.
- **Major Threshold (Red):** Higher threshold that indicates changes from Green or Yellow to Red.

The device also uses hysteresis to determine whether the threshold has indeed being crossed. Hysteresis defines the amount of fluctuation from the threshold in order for the threshold to be considered as crossed (i.e., change in color state). Hysteresis is used to avoid false reports being sent by the device. Hysteresis is used only for threshold crossings toward a lesser severity (i.e., from Red to Yellow, Red to Green, or Yellow to Green).

The following example is used to explain how the device considers threshold crossings. The example is based on the ASR of a call, where the Major threshold is configured to 70%, the Minor threshold to 90% and the hysteresis for both thresholds to 2%:

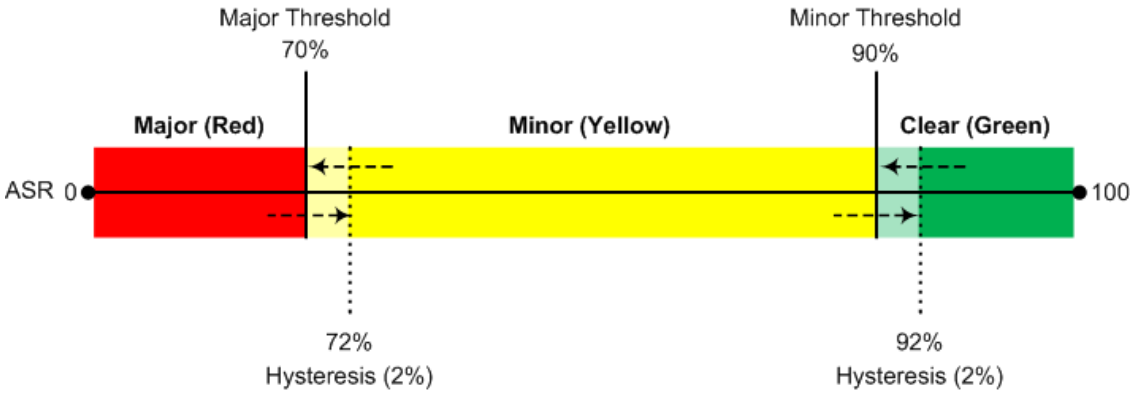


Table 46-1: Threshold Crossings based on Threshold and Hysteresis

Threshold Crossing	Calculation	Threshold based on Example
Green to Yellow (Minor alarm)	The change occurs if the measured metric crosses the configured Minor threshold only (i.e., hysteresis is not used).	90%
Green to Red (Major alarm)	The change occurs if the measured metric crosses the configured Major threshold only (i.e., hysteresis is not used).	70%
Yellow to Red (Major alarm)	The change occurs if the measured metric crosses the configured Major threshold only (i.e., hysteresis is not used).	70%
Red to Yellow (Minor alarm)	The change occurs if the measured metric crosses the configured Major threshold with hysteresis.	72% (i.e., 70 + 2)
Red to Green (alarm cleared)	The change occurs if the measured metric crosses the configured Minor threshold with hysteresis.	92 (i.e., 90 + 2)
Yellow to Green (alarm	The change occurs if the measured metric crosses the configured Minor threshold with hysteresis.	92 (i.e., 90 + 2)

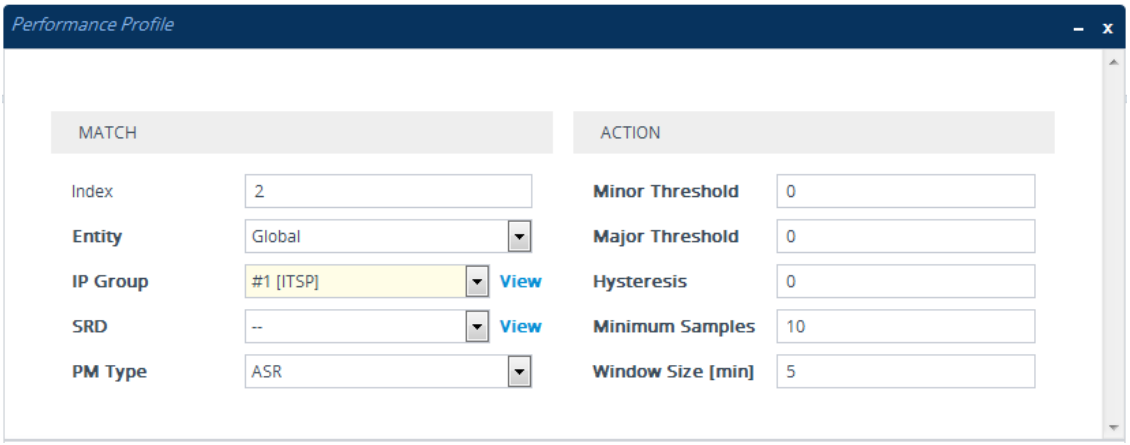
Threshold Crossing	Calculation	Threshold based on Example
cleared)		

- Forwarded calls are not considered in the calculation for ASR and NER.
- If you don't configure thresholds for a specific metric, the device still provides current performance monitoring values of the metric, but does not raise any threshold alarms for it.
- You can configure the device to perform certain actions, for example, reject calls to the IP Group for a user-defined duration, if a threshold is crossed. For more information, see Configuring Quality of Service Rules.

The following procedure describes how to configure Performance Profile rules through the Web interface. You can also configure it through ini file [PerformanceProfile] or CLI (`configure system > performance-profile`).

➢ **To configure a Performance Profile rule:**

1. Open the Performance Profile table (**Monitor** menu > **Monitor** tab > **Performance Monitoring** folder > **Performance Profile**).
2. Click **New**; the following dialog box appears:



3. Configure the rule according to the parameters described in the table below.
4. Click **Apply**.

Table 46-2: Performance Profile Table Parameter Descriptions

Parameter	Description
'Index' [PerformanceProfile_	Defines an index number for the new table row. **Note:** Each row must be configured with a unique index.

Parameter	Description
Index]	
Match	
'Entity' `entity` [PerformanceProfile_Entity]	Defines a configuration entity type to which you want to apply the rule. ■ [0] **Global** = (Default) The device calculates call metrics for all calls. ■ [1] **SRD** = Assigns an SRD. To specify the SRD, use the 'SRD' parameter (see below). ■ [2] **IP Group** = Assigns an IP Group. To specify the IP Group, use the 'IP Group' parameter (see below).
'IP Group' `ip-group-name` [PerformanceProfile_IPGroupName]	Assigns an IP Group to the rule. **Note:** The parameter is applicable only if you configure the 'Entity' parameter to **IP Group**.
'SRD' `srd-name` [PerformanceProfile_SRDName]	Assigns an SRD to the rule. **Note:** The parameter is applicable only if you configure the 'Entity' parameter to **SRD**.
'PM Type' `pmtype` [PerformanceProfile_PMType]	Defines the type of performance monitoring metric for which you want to configure thresholds. ■ [16] **ASR** (Default) ■ [17] **ACD** ■ [18] **NER**
Action	
'Minor Threshold' `minor-threshold` [PerformanceProfile_MinorThreshold]	Defines the Minor threshold (in percentage) of the selected performance monitoring metric, which is the lower threshold located between the Yellow and Green states. To consider a threshold crossing: ■ **Increase in severity (i.e., Green to Yellow):** Only this value is used. ■ **Decrease in severity (Red to Green, or Yellow to Green):** This value is used with the hysteresis, configured by the 'Hysteresis' parameter (see below).

Parameter	Description
	The valid range is 0 to 100. The default is 0.
'Major Threshold' major-threshold [PerformanceProfile_MajorThreshold]	Defines the Major threshold (in percentage) of the selected performance monitoring metric, which is the upper threshold located between the Yellow and Red states. To consider a threshold crossing: ■ **Increase in severity (i.e., Yellow to Red, or Green to Red):** Only this value is used. ■ **Decrease in severity (Red to Yellow):** This value is used with the hysteresis, configured by the 'Hysteresis' parameter (see below). The valid range is 0 to 100. The default is 0.
'Hysteresis' hysteresis [PerformanceProfile_Hysteresis]	Defines the amount of fluctuation (hysteresis) from the configured threshold in order for the threshold to be considered as crossed. Hysteresis is used to avoid false reports being sent by the device. Hysteresis is used only when the severity level decreases (i.e., from Red to Yellow, Yellow to Green, or Red to Green). The valid value is 0 to 15 (in percentage). The default is 5. For example, if you configure the 'Major Threshold' parameter to 70% and the 'Hysteresis' parameter to 2%, the device considers a threshold crossing from Red to Yellow only if the ASR crosses 72% (i.e., 70% + 2%).
'Minimum Samples' minimum-samples [PerformanceProfile_MinimumSample]	Defines the minimum number of call sessions (sample) that is required for the device to calculate the performance monitoring metrics (per window size). If the number of call sessions is less than the configured value, no calculation is done. The default is 10 calls. **Note:** The calculation also depends on the configured sampling window size (see 'Window Size' parameter). For example, if the parameter is configured to 10 calls, but only 5 calls were processed during the configured sampling window, no calculation is done.
'Window Size' window-size [PerformanceProfile_WindowSize]	Defines the time interval (in minutes) during which the device calculates the performance monitoring metrics. For example, if the parameter is configured to five minutes, the calculation is done for the last five minutes.

Parameter	Description
	The default is 5 minutes. **Note:** The calculation depends on the configured minimum samples (see 'Minimum Samples' parameter). For example, if the parameter is configured to five minutes, but the number of calls during the interval is less than the configured minimum samples, no calculation is done.

47 Viewing VoIP Status

This section describes how to view VoIP-related status.

Viewing SBC Registered Users

You can view SBC users that are registered with the device. For each user, the Address of Record (AOR) and the corresponding contacts are shown. An AOR is a SIP or SIPS URI that points to a domain with a location service that can map the URI to another URI (contact) where the user might be available. A contact is a SIP URI that can be used to contact that specific instance of the user agent for subsequent requests.

➢ **To view registered SBC users:**

- Web: SBC Registered Users page (**Monitor** menu > **Monitor** tab > **VoIP Status** folder > **SBC Registered Users**).

ADDRESS OF RECORD	CONTACT
Joe	UserInfo Contact | Not-Active | IPG:1 | SI:-1 | ID:23
Sue	UserInfo Contact | Not-Active | IPG:1 | SI:-1 | ID:24

Table 47-1: SBC Registered Users Table Description

Parameter	Description
Address of Record	Displays the AOR, for example, "1000@10.8.5.71" or "Sue".
Contact	Displays the contacts associated with the AOR, for example: <sip:1000@10.8.5.71:5060>;expires=180; Active status:1 The information displayed can include the following: ■ The origin of the contact: ✓ "Userinfo": Contact is from the User Information table (see Configuring SBC User Information Table through Web Interface on page 590) ✓ Contact URI if the contact is not from the User Information table. ■ Registration status: ✓ No display: The contact has been successfully registered with the device and calls can be routed to the user. ✓ "Not-Active": The device has recently received a REGISTER request from the contact, but the contact has yet to be registered. The device removes the contact from the database if no response is received within 10 seconds from the proxy/registrar server.

Parameter	Description
	■ IP Group to which the contact belongs, shown in the format "IPG:<row index of IP Group)". ■ ID of the contact.

■ CLI:

- SBC users:

```
# show voip register db sbc list
```

- SBC contacts of a specified AOR:

```
# show voip register db sbc user <Address Of Record>
```

Viewing Proxy Set Status

You can view the status of Proxy Sets that are used in your call routing topology. Proxy Sets that are not associated with any routing rule are not displayed. To configure Proxy Sets, see Configuring Proxy Sets.

➢ **To view the status of Proxy Sets:**

■ Open the Proxy Sets Status page (**Monitor** menu > **Monitor** tab > **VoIP Status** folder > **Proxy Sets Status**).

PROXY SET ID	NAME	MODE	KEEP ALIVE	ADDRESS	PRIORITY	WEIGHT	SUCCESS COUNT	FAILURE COUNT	STATUS
0	**ITSP-1**	**Load Balancing**	**Enabled**						**ONLINE**
				10.8.6.88	-	-	0	18	OFFLINE
				10.8.6.89(*)	-	-	3	0	ONLINE
1	**IP-PBX**	**Homing**	**Enabled**						**OFFLINE**
				10.8.6.66	-	-	0	24	
2	**ITSP-2**	**Parking**	**Enabled**						**NOT RESOLVED**
				abc.com	-	-	0	0	NOT RESOLVED

Table 47-2: Proxy Sets Status Table Description

Parameter	Description
Proxy Set ID	Displays the Proxy Set ID.
Name	Displays the Proxy Set name.

Parameter	Description
Mode	Displays the Proxy Sets' operational mode: ■ "Parking" or "Homing": Redundancy mode, as configured by the ProxySet_ProxyRedundancyMode parameter. ■ "Load Balancing: Proxy load balancing mode, as configured by the ProxySet_ProxyRedundancyMode parameter.
Keep Alive	Displays whether the Proxy Keep-Alive feature is enabled ("Enabled") or disabled ("Disabled"), as configured by the ProxySet_ EnableProxyKeepAlive parameter.
Address	Displays the IP address of the proxy server. This can be the IP address as configured in dotted-decimal notation for the Proxy Set, or the resolved IP address of a DNS query if an FQDN is configured for the Proxy Set. ■ IP addresses resolved from FQDNs are displayed as "<FQDN name> (<resolved IP address>)", for example, "abc.com(10.8.6.80)". ■ The IP address that is currently used for routing is indicated with an asterisk, for example, "10.8.6.89(*)". ■ If the FQDN failed to be resolved, only the FQDN name is displayed (e.g., "abc.com").
Priority	Displays the priority of IP addresses resolved from FQDNs. **Note:** The field is applicable only to Proxy Sets configured with FQDNs.
Weight	Displays the weight of IP addresses resolved from FQDNs. **Note:** The field is applicable only to Proxy Sets configured with FQDNs.
Success Count	Displays the total number of successful keep-alive messages (by SIP OPTIONS) sent by the device to the proxy.
Failure Count	Displays the total number of failed keep-alive messages (by SIP OPTIONS) sent by the device to the proxy.
Status	Displays the status of the Proxy Set and its' proxy servers. ■ "ONLINE": ✓ Proxy Set ID row: At least one proxy is online as determined by the device's keep-alive feature. The status is also "ONLINE" for IP addresses resolved from DNS queries even if keep-alive is disabled. ✓ Proxy server rows (if multiple addresses): The proxy server is online as determined by the device's keep-alive feature.

Parameter	Description
	■ "OFFLINE": The proxy is offline as determined by the device's keep-alive feature and the Proxy Set is configured for Homing ('Redundancy Mode' parameter) or enabled for load balancing ('Proxy Load Balancing Method' parameter): ✓ Homing: The proxy is the main proxy, but the keep-alive has failed. ✓ Load balancing: The keep-alive for the proxy has failed. ■ "NOT RESOLVED": Proxy address is configured as an FQDN, but the DNS resolution has failed. ■ Empty field: Keep-alive for the proxy is disabled or the device has yet to send a keep-alive to the proxy.

Viewing Registration Status

You can view registration status of the device's) and SIP Accounts.

➢ **To view registration status:**

■ Open the Registration Status page (**Monitor** menu > **Monitor** tab > **VoIP Status** folder > **Registration Status**).

Accounts Registration Status

INDEX	GROUP TYPE	GROUP NAME	STATUS
1	IP Group		NOT REGISTERED

Table 47-3: Registration Status Page Description

Parameter	Description
Accounts Registration Status	Displays the status registration per Account, as configured in the Accounts table (see Configuring Registration Accounts). ■ **Group Type:** Served IP Group ■ **Group Name:** Name of served IP Group, if applicable ■ **Status:** "REGISTERED" or "NOT REGISTERED"

Viewing CDR History of SBC and Test Calls

You can view historical Call Detail Records (CDR) of SBC calls and Test calls in the SBC CDR History table. History CDRs are stored on the device's memory. When a new CDR is generated, the device adds it to the top of the table and all existing entries are shifted one down in the

table. The table displays the last 4,096 CDRs. If the table reaches maximum capacity of entries and a new CDR is added, the last CDR entry is removed from the table.

- The CDR fields in the table cannot be customized.
- If the device is reset, all CDRs are deleted from memory and from the table.
- You can hide the values of the Caller and Callee fields, as described in Hiding Caller and Callee CDR Field Values on page 1046.

➢ **To view SBC and Test Call CDR history:**

- **Web:** Open the SBC CDR History table (**Monitor** menu > **Monitor** tab > **VoIP Status** folder > **SBC CDR History**).

CALL END TIME	ENDPOINT TYPE	IP GROUP	CALLER	CALLEE	DIRECTION	REMOTE IP	DURATION	TERMINATION REASON	SESSION ID
11:53:30.140 UTC Su	TEST	IPGroup_1	200	200	Incoming	10.33.8.52	00:00:41	NORMAL_CALL_CLEAR	f0000d:1:228
11:52:39.415 UTC Su	TEST	IPGroup_2	201	100	Outgoing	10.33.8.52	00:00:20	NORMAL_CALL_CLEAR	f0000d:1:227
11:52:04.458 UTC Su	SBC	IPGroup_2	200	236	Incoming	10.33.8.52	00:00:02	NORMAL_CALL_CLEAR	f0000d:1:226
11:52:04.444 UTC Su	SBC	IPGroup_2	200	236	Outgoing	10.33.8.52	00:00:02	NORMAL_CALL_CLEAR	f0000d:1:226

- **CLI:**
  - All CDR history:

```
# show voip calls history sbc
```

  - CDR history for a specific SIP session ID:

```
# show voip calls history sbc <session ID>
```

Table 47-4: SBC CDR History Table

Field	Description
Call End Time	Displays the time at which the call ended. The time is displayed in the format, hh:mm:ss, where *hh* is the hour, *mm* the minutes and *ss* the seconds (e.g., 15:06:36).
Endpoint Type	Indicates the type of CDR: - "SBC": CDR belongs to an SBC call. - "TEST": CDR belongs to a Test call.
IP Group	Displays the IP Group of the leg for which the CDR was generated.
Caller	Displays the phone number (source URI user@host) of the party who made the call.

Field	Description
Callee	Displays the phone number (destination URI user@host) of the party to whom the call was made.
Direction	Displays the direction of the call: ■ "Incoming" ■ "Outgoing"
Remote IP	Displays the IP address of the call party. For an "Incoming" call, this is the source IP address; for an "Outgoing" call, this is the destination IP address.
Duration	Displays the duration of the call, displayed in the format hh:mm:ss, where *hh* is hours, *mm* minutes and *ss* seconds. For example, 00:01:20 denotes 1 minute and 20 seconds.
Termination Reason	Displays the reason for the call being released (ended). For example, "NORMAL_CALL_CLEAR" indicates a normal termination.
Session ID	Displays the SIP session ID of the call.

48 Viewing Network Status

This section describes how to view network-related status.

Viewing Active IP Interfaces

You can view the device's active IP interfaces that are configured in the IP Interfaces table (see Configuring IP Network Interfaces).

➢ **To view active IP network interfaces:**

- Open the IP Interface Status page (**Monitor** menu > **Monitor** tab > **Network Status** folder > **IP Interface Status**).

INDEX	APPLICATION TYPE	IP ADDRESS	INTERFACE MODE	PREFIX LENGTH	DEFAULT GATEWAY	INTERFACE NAME	PRIMARY DNS SERVER IP ADDRESS	SECONDARY DNS SERVER IP ADDRESS	UNDERLYING DEVICE	ADDRESS STATE
0	O+M+C	10.15.7.96	IPv4 Manual	16	10.15.0.1	O+M+C	0.0.0.0	0.0.0.0	vlan 1	Permanent
NA	Internal	169.253.254.254	IPv4 Manual	16	0.0.0.0	Internalif 2	0.0.0.0	0.0.0.0	Internalif 2	Permanent

Viewing Ethernet Device Status

You can view the status of configured Ethernet Devices that have been successfully applied. To configure Ethernet Devices, see Configuring Underlying Ethernet Devices.

➢ **To view Ethernet Device status:**

- Open the Ethernet Device Status page (**Monitor** menu > **Monitor** tab > **Network Status** folder > **Ethernet Device Status**).

INDEX	VLAN ID	UNDERLYING INTERFACE	NAME
0	1	GROUP_1	vlan 1
1	2	GROUP_2	vlan 2

Viewing Ethernet Port Information

You can view status information of the device's Ethernet ports. To configure Ethernet ports, see Configuring Underlying Ethernet Devices.

> If the device is operating in High-Availability mode, you can also view Ethernet port information of the redundant device, by opening the Redundant Ethernet Port Information page (**Monitor** menu > **Monitor** tab > **Network Status** folder > **Redundant Ethernet Port Information**).

➢ **To view Ethernet port information:**

- Open the Ethernet Port Information table, by doing one of the following:

- Navigation tree: **Monitor** menu > **Monitor** tab > **Network Status** folder > **Ethernet Port Information.**
- Monitor home page: Click an Ethernet port on the graphical display of the device (see Viewing Device Status on Monitor Page).

Table 48-1: Ethernet Port Information Table Description

Parameter	Description
Port Name	Displays the user-defined name of the port.
Active	Displays if the port is active ("Yes") or not ("No").
Speed	Displays the speed of the Ethernet port.
Duplex Mode	Displays if the port is half- or full-duplex mode.
State	Displays the status of the port. If the Ethernet Group contains a single port, the port status is always "Forwarding" (i.e., data is being transmitted and received on the port). If the Ethernet Group contains two ports (for 1+1 port redundancy): ■ If the 'Mode' of the port is **1RX/1TX**, one port is "Forwarding" (i.e., active port) and one port is "Disabled" (i.e., standby port). ■ If the 'Mode' of the port is **2RX/1TX** or **2RX/2TX**, both ports are "Forwarding".
Group Member	Displays the Ethernet Group to which the port belongs.

Viewing Static Routes Status

You can view the status of static IP routes, configured in the Static Routes table (see Configuring Static IP Routing) and routes through the Default Gateway.

The status of the static routes can be one of the following:

- "Active": Static route is used by the device.
- "Inactive": Static route is not used. When the destination IP address is not on the same segment with the next hop, or the interface does not exist, the route state changes to "Inactive".

➢ **To view the status of static IP routing:**

- Open the Static Route Status table (**Monitor** menu > **Monitor** tab > **Network Status** folder > **Static Route Status**).

INDEX	DESTINATION IP ADDRESS	PREFIX LENGTH	GATEWAY IP ADDRESS	METRIC	DEVICE NAME	STATUS	DESCRIPTION
NA	10.15.0.0	16	0.0.0.0	0	vlan 1	Active	
NA	10.13.0.0	16	0.0.0.0	0	vlan 2	Active	
NA	0.0.0.0	0	10.15.0.1	1	vlan 1	Active	

Viewing IDS Active Blacklist

You can view remote hosts that are currently blacklisted by the device's Intrusion Detection System (IDS) in the IDS Active Black List table. For more information on IDS configuration and blacklists, see Intrusion Detection System on page 187

> For devices in High-Availability (HA) mode, all the table's entries are deleted upon an HA switchover.

The following procedure describes how to view the IDS Active Black List table through the Web interface. You can also view the table through CLI using the command, show voip ids blacklist active.

➢ **To view the active IDS blacklist:**

- Open the IDS Active Black List page (**Monitor** menu > **Monitor** tab > **Network Status** folder > **IDS Active Black List**).

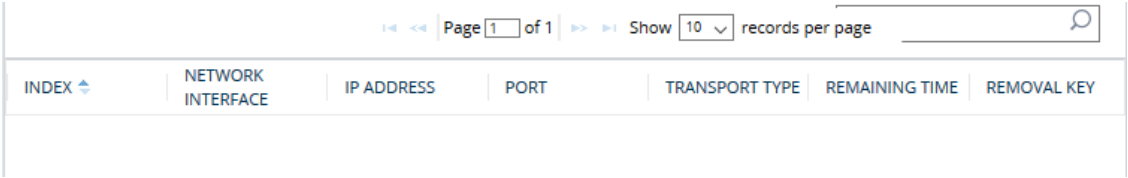


Table 48-2: IDS Active Black List Table Description

Field	Description
Index	Table row index.
Network Interface	The device's IP Interface on which the malicious attack was detected.
IP Address	The IP address of the attacker (remote host).
Port	The port of the attacker (remote host). **Note:** The field is applicable only if the 'Threshold Scope' (IDSRule_ThresholdScope) parameter of the associated IDS rule is configured to **IP+Port**.
Transport Type	The transport type used for the attack.
Remaining Time	The duration left until the device deletes the attacker

Field	Description
	(remote host) from the table and takes it off the IDS blacklist. The blacklisted period is configured by the 'Deny Period' (IDSRule_DenyPeriod) parameter.
Removal Key	A unique number (key) that the device assigns to the blacklisted entry. This is used if you want to remove a specific blacklisted entry from the table, which is done through the CLI command, clear voip ids blacklist <Removal Key>.

49 Viewing Hardware Status

This section describes how to view hardware-related status.

Viewing Hardware Components Status

You can view the status of the device's hardware components such as fans and power supply units on the Components Status page. If the device is in High-Availability mode, the page also displays the status of the hardware components of the Redundant device. The page is refreshed every 10 seconds.

➢ **To view the status of the device's hardware components:**

- Open the Components Status table, by doing one of the following:
  - **Navigation tree: Monitor** menu > **Monitor** tab > **Hardware** folder > **Components Status.**
  - **Monitor home page:** Click a power supply or fan tray icon (see Viewing Device Status on Monitor Page).

ACTIVE BOARD COMPONENTS		REDUNDANT BOARD COMPONENTS	
CPU Slot Number	5	CPU Slot Number	3
Fan	Exists	Fan	Exists
Power Supply no.1	No Alarm	Power Supply	Does not exist
Power Supply no.2	Does not exist	Power Supply no.2	Exists

50 Reporting Information to External Party

This section describes features for reporting various information to an external party.

Configuring RTCP XR

RTP Control Protocol Extended Reports (RTCP XR) is a VoIP management control that defines a set of metrics containing information for assessing VoIP call quality and for diagnosing problems. RTCP XR (RFC 3611) extends the RTCP reports defined in RFC 3550 by providing additional VoIP metrics (Quality of Experience). RTCP XR information publishing is implemented in the device according to RFC 6035. The draft defines how a SIP User Agent (UA) publishes the detailed information to a defined collector. RTCP XR measures VoIP call quality such as packet loss, delay, signal / noise / echo levels, estimated R-factor, and mean opinion score (MOS). RTCP XR measures these parameters using metrics as listed in the table below. RTCP XR messages containing key call-quality-related metrics are exchanged periodically (user-defined) between the device and the SIP UA. This allows an analyzer to monitor these metrics midstream, or a device to retrieve them through SNMP.

- The RTCP XR feature is available only if the device is installed with a License Key that includes this feature. For installing a License Key, see License Key.
- If the RTCP XR feature is unavailable (not licensed or disabled), the R-factor VoIP metrics are not provided in CDRs (CDR fields, Local R Factor and Remote R Factor) generated by the device. Instead, these CDR fields are sent with the value 127, meaning that information is unavailable.
- While the device attempts to determine the signal level, it reports a MOS value of “127 (NA)”. Once it has determined the signal level, it reports the estimated MOS.
- Packet loss effects voice quality estimation only during periods of voice. During periods of silence, packet loss does not effect or degrade voice quality.

You can configure the device to send RTCP XR to a specific IP Group. The device sends RTCP XR in SIP PUBLISH messages. The PUBLISH message contains the following RTCP XR related header values:

- From and To: Telephone extension number of the user
- Request-URI: IP Group name to where RTCP XR is sent
- Event: "vq-rtcpxr"
- Content-Type: "application/vq-rtcpxr"

You can also configure the stage of the call at which you want the device to send RTCP XR:

- End of the call.
- Periodically, according to a user-defined interval between consecutive reports.

The type of RTCP XR report event (VQReportEvent) supported by the device is VQSessionReport (SessionReport). The device can include local and remote metrics in the RTCP XR. Local metrics

are generated by the device while remote metrics are provided by the remote endpoint. The following table lists the supported voice metrics (parameters) published in the RTCP XR.

Table 50-1: RTCP XR Published VoIP Metrics

Metric	Parameter	Description
CallID	-	Call ID - call ID from the SIP dialog
LocalID	-	Local ID - identifies the reporting endpoint for the media session
RemoteID	-	Remote ID - identifies the remote endpoint of the media session
OrigID	-	Originating ID - Identifies the endpoint which originated the session
LocalAddr	-	Local Address - IP address, port, and SSRC of the endpoint/UA which is the receiving end of the stream being measured
RemoteAddr	-	Remote Address - IP address, port, and SSRC of the the source of the stream being measured
LocalGroup	-	Local Group ID - identification for the purposes of aggregation for the local endpoint
RemoteGroup	-	Remote Group ID - identification for the purposes of aggregation for the remote endpoint
LocalMAC	-	Media Access Control (MAC) address of the local SIP device
Timestamps	START	Start time of the media session.
	STOP	End time of the media session.
SessionDesc	PT	Payload Type - 'payload type' parameter in the RTP packets (i.e., the codec).
	PD	Payload Description - description of the codec
	SR	Sample Rate - rate at which the voice was sampled
	FD	Frame Duration (msec) - packetization rate
	FO	Frame Octets - number of octets in each frame

Metric	Parameter	Description
		per RTP packet
	FPP	Frames per Packets - number of frames per RTP packet
	PLC	Packet Loss Concealment - indicates whether a PLC algorithm was used for the session ("0" - unspecified; "1" - disabled; "2" - enhanced; "3" - standard)
	SSUP	Silence Suppression State - indicates whether silence suppression, also known as Voice Activity Detection (VAD) is enabled ("on" or "off")
JitterBuffer	JBA	Jitter Buffer Adaptive - indicates the jitter buffer in the endpoint ("0" - unknown; "1" - reserved; "2" - non-adaptive; "3" - adaptive)
	JBR	Jitter Buffer Rate
	JBN	Jitter Buffer Nominal
	JBM	Jitter Buffer Max
	JBX	Jitter Buffer Abs Max
PacketLoss	NLR	Network Packet Loss Rate
	JDR	Jitter Buffer Discard Rate
BurstGapLoss	BLD	Burst Loss Density
	BD	Burst Duration
	GLD	Gap Loss Density
	GD	Gap Duration
	GMIN	Minimum Gap Threshold

Metric	Parameter	Description
Delay	RTD	Round Trip Delay (msec)
	ESD	End System Delay (msec)
	OWD	One Way Delay (msec)
	IAJ	Inter-Arrival Jitter (msec)
	MAJ	Mean Absolute Jitter (msec)
Signal	SL	Signal Level (dB) - ratio of the signal level to a 0 dBm0 reference
	NL	Noise Level (dB) - ratio of the silent period background noise level to a 0 dBm0 reference
	RERL	Residual Echo Return Noise (dB) - ratio between the original signal and the echo level as measured after echo cancellation or suppression has been applied
QualityEst	RLQ	Listening Quality R - listening quality expressed as an R factor (0-95 for narrowband calls and 0-120 for wideband calls)
	RLQEstAlg	RLQ Est. Algorithm - name (string) of the algorithm used to estimate RLQ
	RCQ	Conversational Quality R - cumulative measurement of voice quality from the start of the session to the reporting time (R is 0-95 for narrowband calls and 0-120 for wideband calls)
	RCQEstAlg	RCQ Est. Algorithm - name (string) of the algorithm used to estimate RCQ
	EXTRI	External R In - voice quality as measured by the local endpoint for incoming connection on "other" side (R is 0-95 for narrowband calls and 0-120 for wideband calls)
	ExtRIEstAlg	Ext. R In Est. Algorithm - name (string) of the algorithm used to estimate EXTRI
	EXTRO	External R Out - value is copied from RTCP XR received from the remote endpoint on the "other"

Metric	Parameter	Description
		side of this endpoint (R is 0-95 for narrowband calls and 0-120 for wideband calls)
	ExtROEstAlg	Ext. R Out Est. Algorithm - name (string) of the algorithm used to estimate EXTRO
	MOSLQ	MOS-LQ - estimated mean opinion score for listening voice quality on a scale from 1 to 5, in which 5 represents excellent and 1 represents unacceptable
	MOSLQEstAlg	MOS-LQ Est. Algorithm - name (string) of the algorithm used to estimate MOSLQ
	MOSCQ	MOS-CQ - estimated mean opinion score for conversation voice quality on a scale from 1 to 5, in which 5 represents excellent and 1 represents unacceptable
	MOSCQEstAlg	MOS-CQ Est. Algorithm - name (string) of the algorithm used to estimate MOSCQ
	QoEEstAlg	QoE Est. Algorithm - name (string) of the algorithm used to estimate all voice quality metrics
DialogID	-	Identification of the SIP dialog with which the media session is related

Below shows an example of a SIP PUBLISH message sent with RTCP XR and QoE information:

```
PUBLISH sip:172.17.116.201 SIP/2.0
Via: SIP/2.0/UDP 172.17.116.201:5060;branch=z9hG4bKac2055925925
Max-Forwards: 70
From: <sip:172.17.116.201>;tag=1c2055916574
To: <sip:172.17.116.201>
Call-ID: 20559160721612201520952@172.17.116.201
CSeq: 1 PUBLISH
Contact: <sip:172.17.116.201:5060>
Allow:
REGISTER,OPTIONS,INVITE,ACK,CANCEL,BYE,NOTIFY,PRACK,REFER,INFO,SUBSCRIBE,UPDATE
Event: vq-rtcpxr
Expires: 3600
User-Agent: device/7.20A.258.980
```

```
Content-Type: application/vq-rtcpxr
Content-Length: 1066
```

```
VQSessionReport
CallID=20328634741612201520943@172.17.116.201
LocalID: <sip:1000@172.17.116.201>
RemoteID: <sip:2000@172.17.116.202;user=phone>
OrigID: <sip:1000@172.17.116.201>
LocalAddr: IP=172.17.116.201 Port=6000 SSRC=0x54c62a13
RemoteAddr: IP=172.17.116.202 Port=6000 SSRC=0x243220dd
LocalGroup:
RemoteGroup:
LocalMAC: 00:90:8f:57:d9:71
```

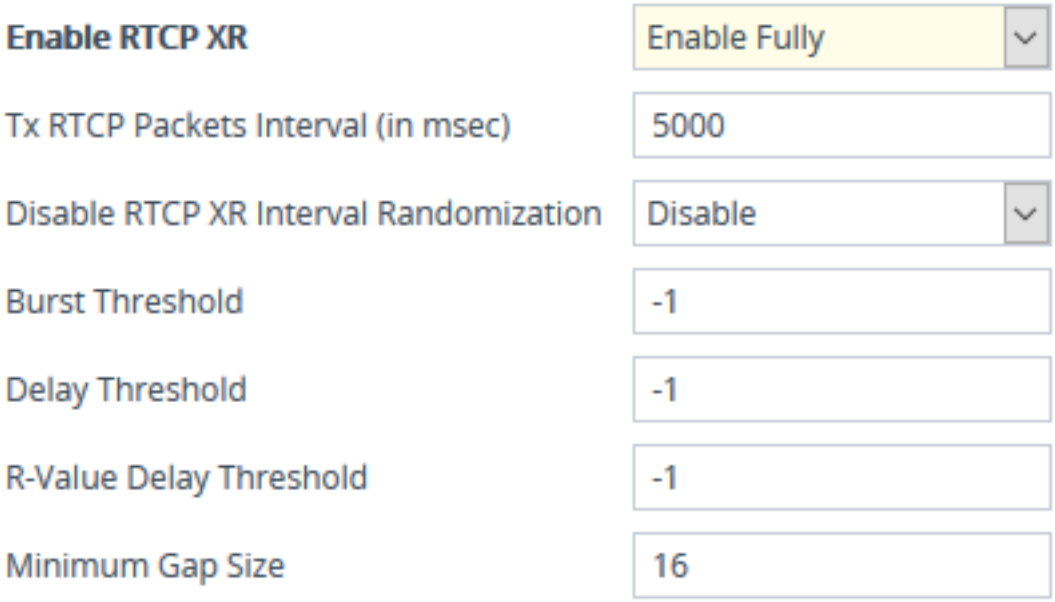
```
LocalMetrics:
Timestamps: START=2015-12-16T20:09:45Z STOP=2015-12-16T20:09:52Z
SessionDesc: PT=8 PD=PCMA SR=8000 FD=20 PLC=3 SSUP=Off
JitterBuffer: JBA=3 JBR=0 JBN=7 JBM=10 JBX=300
PacketLoss: NLR=0.00 JDR=0.00
BurstGapLoss: BLD=0.00 BD=0 GLD=0.00 GD=6325 GMIN=16
Delay: RTD=0 ESD=11
Signal: SL=-34 NL=-67 RERL=17
QualityEst: RLQ=93 MOSLQ=4.1
MOSCQ=4.10
```

```
RemoteMetrics:
Timestamps: START=2015-12-16T20:09:45Z STOP=2015-12-16T20:09:52Z
JitterBuffer: JBA=3 JBR=0 JBN=0 JBM=0 JBX=300
PacketLoss: NLR=0.00 JDR=0.00
BurstGapLoss: BLD=0.00 BD=0 GLD=0.00 GD=0 GMIN=16
Delay: RTD=65535 ESD=0
QualityEst:
DialogID: 20328634741612201520943@172.17.116.201;to-
tag=1c1690611502;from-tag=1c2032864069
```

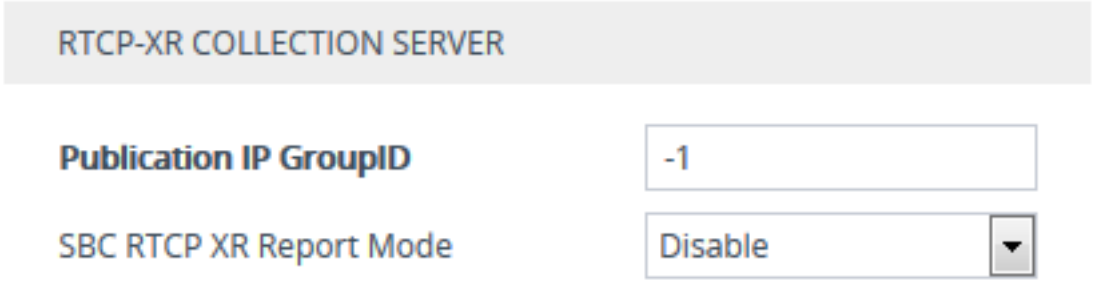
➢ **To configure RTCP XR:**

1. Open the RTP/RTCP Settings page (**Setup** menu > **Signaling & Media** tab > **Media** folder > **RTP/RTCP Settings**).
2. Under the RTCP-XR group, configure the following:
   - 'Enable RTCP XR' (*VQMonEnable*) - enables voice quality monitoring and RTCP XR.

- 'Tx RTCP Packets Interval' (*RTCPInterval*) - defines the time interval between adjacent RTCP reports.
- 'Disable RTCP XR Interval Randomization' (*DisableRTCPRandomize*) - determines whether RTCP report intervals are randomized or whether each report interval accords exactly to the parameter RTCPInterval.
- 'Burst Threshold' (*VQMonBurstHR*) - defines the voice quality monitoring excessive burst alert threshold.
- 'Delay Threshold' (*VQMonDelayTHR*) - defines the voice quality monitoring excessive delay alert threshold.
- 'R-Value Delay Threshold' (*VQMonEOCRValTHR*) - defines the voice quality monitoring end of call low quality alert threshold.
- 'Minimum Gap Size' (*VQMonGMin*) - defines the voice quality monitoring minimum gap size (number of frames).



3. Under the RTCP-XR Collection Server group, configure the following:
   - 'Publication IP Group ID' (PublicationIPGroupID): Configures the IP Group to where you want the device to send RTCP XR reports.
   - 'SBC RTCP XR Report Mode' (SBCRtcpXrReportMode): Enables the sending of RTCP XR reports of QoE metrics at the end of each call session (i.e., after a SIP BYE).



4. Click **Apply**, and then reset the device with a save-to-flash for your settings to take effect.

Call Detail Records

Call Detail Records (CDR) contains vital statistic information on calls made from the device. The device can generate and report CDRs at various stages of the call - end of call, or only at the start and end of call. In addition, CDRs can be generated for SIP signaling and/or media. The device can send CDRs to any of the following:

- Syslog server. The CDR Syslog message complies with RFC 3164 and is identified by Facility 17 ("local1") and Severity 5 (Notice).
- RADIUS server. For CDR in RADIUS format, see Configuring RADIUS Accounting. To configure RADIUS servers for CDR reporting, see Configuring RADIUS Servers.

- To view SBC and Test Call CDRs stored on the device's memory, see Viewing SBC CDR History.

Enabling CDR Generation and Configuring CDR Server Address

For the device to generate CDRs, you need to enable the Syslog feature and configure a collecting server address. The collecting server can be a dedicated CDR server or the server used for Syslog messages.

➢ **To enable CDR generation and configure the CDR server address:**

1. Enable the Syslog feature (see Enabling Syslog).
2. Open the Call Detail Record Settings page (**Troubleshoot** menu > **Troubleshoot** tab > **Call Detail Record** folder > **Call Detail Record Settings**).
3. In the 'CDR Syslog Server IP Address' [CDRSyslogServerIP] field, enter the IP address of the server to where you want the CDRs sent.

4. Click **Apply**.

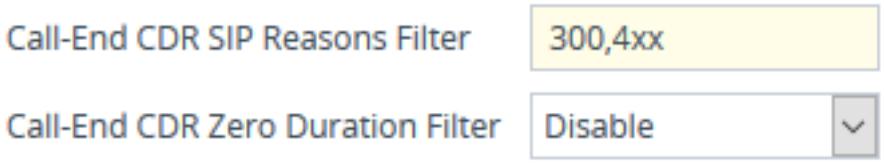
- If you do not configure an IP address for a CDR server, the device sends the CDRs to the address that you configured for the Syslog server (see Configuring the Syslog Server Address on page 1073).
- The port configured for the Syslog server is also used for the CDR server (see Configuring the Syslog Server Address on page 1073).

Configuring CDR Filters and Report Level

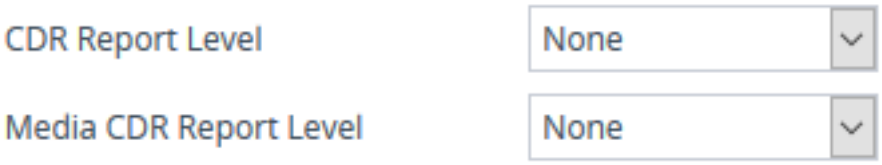
You can configure various CDR filters and the stage of the call at which you want the device to generate and send CDRs. For a detailed description of the parameters described in this section, see Syslog, CDR and Debug Parameters.

➢ **To configure CDR filters and report level:**

1. Open the Call Detail Record Settings page (**Troubleshoot** menu > **Troubleshoot** tab > **Call Detail Record** folder > **Call Detail Record Settings**).
2. Configure CDR filters:
   - In the 'Call-End CDR SIP Reasons Filter' [CallEndCDRSIPReasonsFilter] field, configure the SIP release cause codes that if received for calls, the device does not generate and send Call-End CDRs.
   - From the 'Call-End CDR Zero Duration Filter' [CallEndCDRZeroDurationFilter] drop-down list, select **Enable** if you want the device to not generate and send Call-End CDRs for calls of zero (0) duration.



3. Configure the call stage at which CDRs are generated and sent:
   - From the 'CDR Report Level' [CDRReportLevel] drop-down list, select the stage of the call at which you want signaling-related CDRs to be generated and sent.
   - From the 'Media CDR Report Level' [MediaCDRReportLevel] drop-down list, select the stage of the call at which you want media-related CDRs to be generated and sent.



4. Click **Apply**.

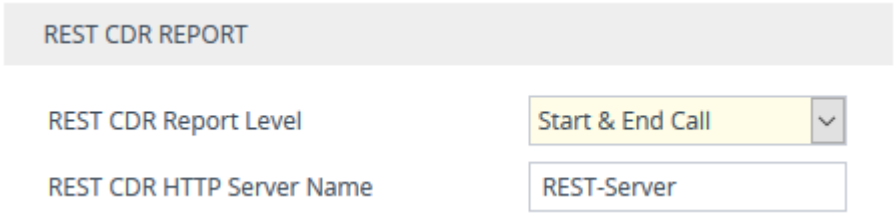
Configuring CDR Reporting to REST Server

You can configure the device to send signaling-related CDRs to a REST server using AudioCodes REST API. The CDRs are sent in JSON format.

> You can customize the CDRs that are sent to the REST server, by adding CDR fields or changing their names. For more information, see Customizing CDRs for SBC Calls and Test Calls on page 1036.

➢ **To configure CDR reporting to a REST server:**

1. Enable the Syslog feature for sending log messages (CDRs) generated by the device to a collecting log message server. For more information, see Enabling Syslog.
2. Configure the REST server:
   a. Open the Remote Web Services table (see Configuring Remote Web Services on page 316).
   b. Click **New**, and then configure an HTTP/S-based server to represent the REST server. Make sure that you configure the 'Type' parameter to **General**. Configure the remaining HTTP/S server parameters according to your requirements.
   c. Click **Apply**.
3. Open the Call Detail Record Settings page (**Troubleshoot** menu > **Troubleshoot** tab > **Call Detail Record** folder > **Call Detail Record Settings**), and then do the following:
   a. From the 'REST CDR Report Level' drop-down list, select the stage of the call at which you want the CDRs to be generated and sent.
   b. In the 'REST CDR HTTP Server Name' field, enter the name of the REST server that you configured in the Remote Web Services table (see Step 2). This is the server where the device sends the CDRs.



   c. Click **Apply**.

Miscellaneous CDR Configuration

Miscellaneous but important CDR configuration parameters are listed below:

- To enable or disable the inclusion of the sequence number (S=) in CDR Syslog messages, use the 'CDR Syslog Sequence Number' [CDRSyslogSeqNum] parameter.
- The device sends CDRs only for dialog-initiating INVITE messages (call start), 200 OK responses (call connect) and BYE messages (call end). If you want to enable the generation of CDRs for non-call SIP dialogs (such as SUBSCRIBE, OPTIONS, and REGISTER), use the [EnableNonCallCdr] parameter.

- To configure the units of measurement for the call duration in CDRs ("Duration" CDR field), use the [CallDurationUnits] parameter.
- To configure the time zone (e.g., GMT+1) that is displayed with the timestamp in CDRs ("Connect Time", "Release Time", and "Setup Time" CDR fields), use the [TimeZoneFormat] parameter

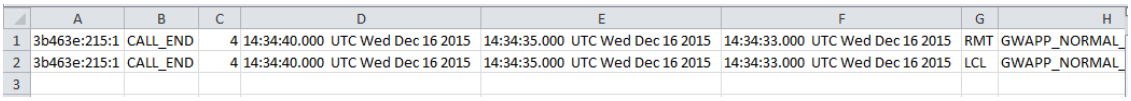
Storing CDRs on the Device

CDRs generated by the device can be stored locally on the device (or internal SD card).

You can specify the calls for which you wish to create locally stored CDRs. This is done using Logging Filter rules in the Logging Filters table. For example, you can configure a rule to create locally stored CDRs for traffic belonging only to IP Group #2.

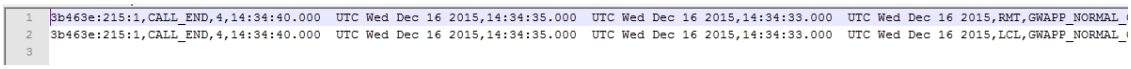
Locally stored CDRs are saved in a comma-separated values file (*.csv), where each CDR is on a dedicated row or line. An example of a CSV file with two CDRs are shown below :

- CSV file viewed in Excel:



	A	B	C	D	E	F	G	H
1	3b463e:215:1	CALL_END	4	14:34:40.000 UTC Wed Dec 16 2015	14:34:35.000 UTC Wed Dec 16 2015	14:34:33.000 UTC Wed Dec 16 2015	RMT	GWAPP_NORMAL_
2	3b463e:215:1	CALL_END	4	14:34:40.000 UTC Wed Dec 16 2015	14:34:35.000 UTC Wed Dec 16 2015	14:34:33.000 UTC Wed Dec 16 2015	LCL	GWAPP_NORMAL_
3								

- CSV file viewed in a text editor (Notepad):



```
3b463e:215:1,CALL_END,4,14:34:40.000  UTC Wed Dec 16 2015,14:34:35.000  UTC Wed Dec 16 2015,14:34:33.000  UTC Wed Dec 16 2015,RMT,GWAPP_NORMAL_
3b463e:215:1,CALL_END,4,14:34:40.000  UTC Wed Dec 16 2015,14:34:35.000  UTC Wed Dec 16 2015,14:34:33.000  UTC Wed Dec 16 2015,LCL,GWAPP_NORMAL_
```

The device's CLI provides enhanced support for performing various actions on locally stored CDRs:

- To view the CDR column headers corresponding to the CDR data in the CSV file:
  - SBC CDRs:

```
(config-system)# cdr
(cdr)# cdr-format show-title local-storage-sbc
session id,report type,call duration, call end time, call connect time,call
start time, call originator, termination reason, call id, srce uri, dest uri
```

- To view stored CDR files:
  - View all stored CDR files:

```
# show storage-history
```

  - View all stored, unused CDR files:

```
# show storage-history unused
```

- To delete stored CDR files:
  - To delete all stored files:

```
# clear storage-history cdr-storage-history all
```

  - To delete all unused stored CDR files:

```
# clear storage-history cdr-storage-history unused
```

- To copy stored CDR files to a remote destination:

```
# copy storage-history cdr-storage-history <filename> to
<protocol://destination>
```

Where:

- *filename:* name you want to assign the file. Any file extension name can be used, but as the file content is in CSV format, it is recommended to use the .csv file extension.
- *protocol:* protocol over which the file is sent (tftp, http, or https).

For example:

```
copy storage-history cdr-storage-history my_cdrs.csv to
tftp://company.com/cdrs
```

- The SD card provides storage capacity of up to 4 GB for Mediant 4000 and 16 GB for Mediant 4000B.
- If the device is reset or powered off, locally stored CDRs are deleted.
- Locally stored CDRs are applicable only to "CALL_END" CDR Report Types and to SBC signaling CDRs.
- When the device operates in High-Availability (HA) mode, active and redundant devices maintain their own stored CDRs. In other words, upon an HA switchover the stored CDRs on the active device are not copied to the redundant device (which becomes the new active device). However, you can view the stored CDRs on the redundant device without performing a switchover, by accessing the redundant device from the active device through SSH (or SFTP), as described in Accessing Files on Redundant Device from Active through SSH on page 847.
- You can customize the CDR fields for local storage. For SBC calls, see Customizing CDRs for SBC Calls.

➢ **To configure local CDR storage:**

1. Open the Logging Filters table (see Configuring Log Filter Rules), and then enable CDR local storage by configuring a log filtering rule with the following settings:
   - 'Filter Type' and 'Value': (as desired)

- 'Log Destination': **Local Storage**
- 'Log Type': **CDR Only**
- 'Mode': **Enable**

2. Open the Call Detail Record Settings page (**Troubleshoot** menu > **Troubleshoot** tab > **Call Detail Record** folder > **Call Detail Record Settings**), and then configure the following parameters:
   - 'File Size' [CDRLocalMaxFileSize]: Enter the maximum size (in kilobytes) of the CDR file. When the Current file reaches this size, the device creates a CDR file. However, if the 'Rotation Period' is reached before the file has reached this maximum size, the CDR file is created.
   - 'Number of Files' [CDRLocalMaxNumOfFiles]: Enter the maximum number of CDR files. If this maximum is reached, any new CDR file replaces the oldest CDR file (i.e., FIFO).
   - 'Rotation Period' [CDRLocalInterval]: Enter the periodic duration (in minutes) of how often a CDR file is created from the Current file (even if empty). For example, if configured to 60, a file is created every hour (or before, if the maximum size has been reached).

   For a detailed description of each parameter, see Syslog, CDR and Debug Parameters.

> If the CDR storage feature is enabled and you later change the maximum number of files ([CDRLocalMaxNumOfFiles]) to a lower value (e.g., from 50 to 10), the device stores the remaining files (e.g., 40) in its memory as *unused* files.

CDR Field Description

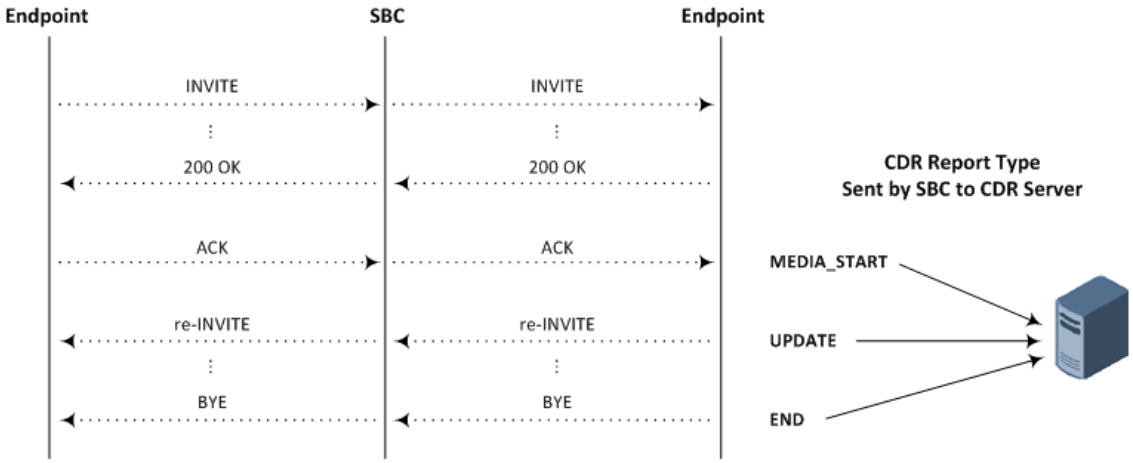
This section describes the CDR fields that can be generated by the device. Some are generated by default while others are generated only if you customize the CDR to include them, as described in Customizing CDRs for SBC Calls.

- For SBC calls, the device generates a signaling CDR and a media CDR:
  - **Media CDR:** This CDR is published per active media stream. Each media CDR has a unique call ID that corresponds to its signaling CDR. There are three different CDR Report Types (CDRReportType), which the device sends to the CDR server at different stages of the SIP dialog session:
    - "MEDIA_START": This CDR is sent upon an INVITE message.

- "MEDIA_UPDATE": This CDR is sent upon a re-INVITE message (e.g., the established call is placed on hold by one of the call parties).
- "MEDIA_END": This CDR is sent upon a BYE message (i.e., call ends).

The CDR Report Types for SBC media and the SIP dialog stages at which they are sent are shown in the following figure:

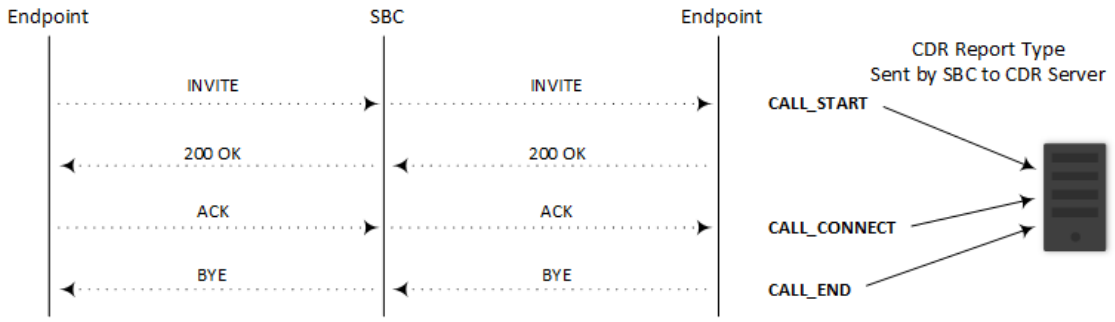
Figure 50-1: CDR Report Types for SBC Media



- **Signaling CDR:** This CDR contains SIP signaling information. A typical SBC session consists of two SBC legs. Each leg generates its own signaling CDRs. Each leg generates three different CDR Report Types (CDRReportType), which the device sends to the CDR server at different stages of the SIP dialog:
  - "CALL_START": This CDR is sent upon an INVITE message.
  - "CALL_CONNECT": This CDR is sent upon an ACK message (i.e., call is established).
  - "CALL_END": This CDR is sent upon a BYE message (i.e., call ends).

  The CDR Report Types for SBC signaling and the SIP dialog stages at which they are sent are shown in the following figure:

Figure 50-2: CDR Report Types for SBC Signaling



You can customize the signaling CDR that is sent at the end of the SBC call ("CALL_END") to also include media-related CDR fields. This is applicable only to syslog CDRs, local storage CDRs, and RADIUS CDRs. For customizing SBC CDRs, see Customizing CDRs for SBC Calls. When there is more than one media stream in the SBC session, the added media-related fields only represent the first audio media.

CDRs belonging to the same SBC session (both incoming and outgoing legs) have the same Session ID (SessionId CDR field). CDRs belonging to the same SBC leg have the same Leg ID (LegId CDR field).

For billing applications, the CDR that the device sends at the end of the call ("CALL_END" CDR Report Type) is usually sufficient. This CDR may be based on the following CDR fields:

- Leg ID
- Source URI
- Destination URI
- Call originator (i.e., caller)
- Call duration
- Call time

The Syslog displays CDRs in tabular format, whereby the CDR field names (titles) are displayed on the first lines and their corresponding values on the subsequent lines. Below shows an example of an SBC signaling CDR sent at the end of a normally terminated call:

```
[S=40] |CDRReportType |EPTyp |SIPCallId |SessionId |Orig |SourceIp |SourcePort
|DestIp |DestPort |TransportType |SrcURI |SrcURIBeforeMap |DstURI
|DstURIBeforeMap |Durat |TrmSd |TrmReason |TrmReasonCategory |SetupTime
|ConnectTime |ReleaseTime |RedirectReason |RedirectURINum
|RedirectURINumBeforeMap |TxSigIPDiffServ|IPGroup (description) |SrdId (name)
|SIPInterfaceId (name) |ProxySetId (name) |IpProfileId (name) |MediaRealmId
(name) |DirectMedia |SIPTrmReason |SIPTermDesc |Caller |Callee
```

```
[S=40] |CALL_END |SBC |207675932914102010170290@10.33.45.80
|1871197419|LCL |10.33.45.80 |5060 |10.33.45.72 |5060 |UDP |9001@10.8.8.10
|9001@10.8.8.10 |6001@10.33.45.80 |6001@10.33.45.80 |15 |LCL |GWAPP_
NORMAL_CALL_CLEAR |NORMAL_CALL_CLEAR |17:00:29.954  UTC Thu Oct
14 2014 |17:00:49.052  UTC Thu Oct 14 2014 |17:01:04.953  UTC Thu Oct 14 2014
|-1 | | |40 |1 |0 (SRD_GW) |1 |1 |1 () |0 (MR_1) |no |BYE |Q.850 ;cause=16 ;text="loc
|user 9928019 |
```

If all CDR field values are within a specific number of characters, they appear aligned under their corresponding field names. However, if some of the values exceed their specific number of characters for Syslog tabular alignment, the values do not appear fully aligned with their

corresponding field names. If you customize the title of a CDR field and it contains more characters than the default title, the maximum number of characters to ensure Syslog tabular alignment will be updated accordingly to fit the customized title. For example, if you customize the default CDR field title "Duration" (8 characters) to "Duration in Sec" (15 characters), the tabular alignment of field names to corresponding values will be updated to 15 as well. The maximum number of characters for Syslog tabular alignment when CDR field titles are not customized are given in the table below.

Table 50-2: CDR Field Descriptions

Field	Description
Accounting Status Type [305]	Displays the CDR Report Type in numeric representation (integer), used mainly for the RADIUS Accounting Status Type attribute (40): ■ "1" = “Accounting Start” for "CALL_START" or "CALL_CONNECT" ■ "2" = “Accounting Stop” for "CALL_END" **Note:** ■ By default, the field is included in the CDR. ■ The field is applicable to SBC media and signaling, and Gateway CDRs (all CDR Report Types). ■ The maximum number of characters for Syslog tabular alignment is 5.
Alerting Time [443]	Displays the duration (in milliseconds) between ringing (SIP 180 Ringing) and call answered (SIP 200 OK) or unanswered (CANCEL). **Note:** ■ The field is optional. You can include it in the CDR by CDR customization using the SBC CDR Format and Gateway CDR Format tables. ■ The field is applicable only to SBC signaling and Gateway CDRs ("CALL_CONNECT" and "CALL_END" Report Types). ■ The maximum number of characters for Syslog tabular alignment is 5.
AMD Decision Probability [630]	Displays the success (in percentage) that the answering type (probability) was correctly detected for the Answering Machine Detection (AMD) feature. **Note:** ■ By default, the field is included in the CDR. ■ The field is applicable only to SBC media CDRs ("MEDIA_END" CDR Report Type) and Gateway CDRs ("CALL_END" CDR Report Type).

Field	Description
	■ The default field title is "%" for Syslog. ■ The maximum number of characters for Syslog tabular alignment is 3.
AMD Decision [629]	Displays the detected answering type for the AMD feature: ■ "V": voice ■ "A": answer machine ■ "S": silence ■ "U": unknown **Note:** ■ By default, the field is included in the CDR. ■ The field is applicable only to SBC media CDRs ("MEDIA_END" CDR Report Type) and Gateway CDRs ("CALL_END" CDR Report Type). ■ The default field title is "AMD" for Syslog. ■ The maximum number of characters for Syslog tabular alignment is 3.
Blank [308]	Displays an empty string value " " and 0 for an integer value. This is typically used for RADIUS CDRs. **Note:** ■ The field is optional. You can include it in the CDR by CDR customization using the SBC CDR Format and Gateway CDR Format tables. ■ The field is applicable to all CDR Report Types. ■ The maximum number of characters for Syslog tabular alignment is 5.
Call Duration [408]	Displays the duration of the call. The field is an integer. **Note:** ■ To configure the units of measurement (seconds - default, deciseconds, centiseconds, or milliseconds), use the [CallDurationUnits] parameter. ■ By default, the field is included in the CDR. ■ The field is applicable only to SBC signaling and Gateway CDRs ("CALL_END" CDR Report Type).

Field	Description
	■ The default field title is "Duration" for Syslog and Local Storage, and none for RADIUS (ACCT_SESSION_TIME standard ID 46). ■ The maximum number of characters for Syslog tabular alignment is 8.
Call End Sequence Number [442]	Displays the sequence number of the call. The field is an integer. For each call-end CDR, the field is assigned the next consecutive number. For example, for the first terminated call processed by the device, the field is assigned the value "1"; for the second terminated call, the field is assigned the value "2", and so on. The field value resets to 1 upon a device reset, an HA switchover (for HA-supporting products), or when it reaches the value FFFFFFFF (hexadecimal). As the field is consecutive, you can use this field to check whether there are any missing CDRs. **Note:** ■ The field is optional. You can include it in the CDR by CDR customization using the SBC CDR Format and Gateway CDR Format tables. ■ The field is applicable only to SBC signaling and Gateway CDRs ("CALL_END" CDR Report Type). ■ The maximum number of characters for Syslog tabular alignment is 10.
Call ID [301]	Displays the unique ID of the call, which appears in the SIP Call-ID header. The field is a string of up to 130 characters. **Note:** ■ By default, the field is included in the CDR. ■ The field is applicable to SBC media and signaling, and Gateway CDRs (all CDR Report Types). ■ The default field title is "SIPCallId" for Syslog and Local Storage, and "call-id=" for RADIUS. ■ The maximum number of characters for Syslog tabular alignment is 50.
Call Orig RADIUS [434]	Displays the originator of the call: ■ "answer": Call originated from the IP side (Gateway) or incoming leg (SBC) ■ "originate": Call originated from the Tel side (Gateway) or outgoing

Field	Description
	leg (SBC) **Note:** ■ By default, the field is included in the CDR. ■ The field is applicable to CDR Report Types "Start Acc" and "Stop Acc". ■ The field is applicable to all types, but mainly to RADIUS (SBC and Gateway CDRs). ■ The default field title is "h323-call-origin=" for RADIUS. ■ The maximum number of characters for Syslog tabular alignment is 10.
Call Orig [401]	Displays which side originated the call for the specific leg. ■ "LCL": SBC Outgoing leg (called party side) or Tel side ■ "RMT": SBC Incoming leg (i.e., caller party side) or IP side **Note:** ■ By default, the field is included in the CDR. ■ The field is applicable only to SBC signaling and Gateway CDRs (all CDR Report Types). ■ The default field title is "Orig" for Syslog and "Direction" in the Web SBC CDR History and Web Gateway CDR History tables. ■ The maximum number of characters for Syslog tabular alignment is 5.
Callee Display ID [432]	Displays the name of the called party. The field is a string of up to 36 characters. **Note:** ■ By default, the field is included in the CDR. ■ The field is applicable only to SBC signaling and Gateway CDRs (all CDR Report Types). ■ The default field title is "Callee" in the sent CDR. ■ The maximum number of characters for Syslog tabular alignment is 37.
Caller Display ID [431]	Displays the name of the caller (caller ID). The field is a string of up to 50 characters.

Field	Description
	Note: ■ By default, the field is included in the CDR. ■ The field is applicable only to SBC signaling and Gateway CDRs (all CDR Report Types). ■ The default field title is "Caller" in the CDR. ■ The maximum number of characters for Syslog tabular alignment is 51.
CDR Type [300]	Displays the application type of the CDR. The field is an integer: ■ "2": Gateway CDR ■ "3": SBC signaling CDR ■ "4": SBC media CDR **Note:** ■ The field is optional. You can include it in the CDR by CDR customization using the Gateway CDR Format and SBC CDR Format tables. ■ The field is applicable only to SBC media and signaling, and Gateway CDRs (all CDR Report Types). ■ The maximum number of characters for Syslog tabular alignment is 6.
Channel ID [600]	Displays the port (channel) ID. **Note:** ■ By default, the field is included in the CDR. ■ The field is applicable only to SBC media CDRs ("MEDIA_START" and "MEDIA_END" CDR Report Types) and Gateway CDRs ("CALL_START", "CALL_CONNECT" and "CALL_END" CDR Report Types). ■ The default field title is "Cid" in the CDR. ■ The maximum number of characters for Syslog tabular alignment is 5.
Coder Type [601]	Displays the coder used for the call. The field is a string, for example, "g711Alaw64k", "g711Ulaw64k" and "g729". **Note:** ■ By default, the field is included in the CDR.

Field	Description
	■ The field is applicable only to SBC media CDRs ("MEDIA_START" and "MEDIA_END" CDR Report Types) and Gateway CDRs ("CALL_CONNECT" and "CALL_END" CDR Report Types). ■ The default field title is "Coder" in the CDR. ■ The maximum number of characters for Syslog tabular alignment is 15.
Connect Time [412]	Displays the date and time that the call was connected. The field is a string of up to 35 characters and in the following format: <hh:mm:ss:ms> UTC <DDD> <MMM> <DD> <YYYY>. For example, "17:00:49.053 UTC Thu Dec 14 2017" **Note:** ■ To configure the time zone string (e.g., "UTC" - default, "GMT+1" and "EST"), use the TimeZoneFormat parameter. ■ By default, the field is included in the CDR. ■ The field is applicable only to SBC signaling and Gateway CDRs ("CALL_CONNECT" and "CALL_END" CDR Report Types). ■ The default field title is "ConnectTime" for Syslog and Local Storage, and "h323-connect-time=" for RADIUS. ■ The maximum number of characters for Syslog tabular alignment is 35.
Dest Port [406]	Displays the SIP signaling destination UDP port. The field is an integer of up to 10 digits. **Note:** ■ By default, the field is included in the CDR. ■ The field is applicable only to SBC signaling and Gateway CDRs (all CDR Report Types). ■ The default field title is "SigDestPort" for Gateway Syslog and Local Storage, and "DestPort" for SBC Syslog and Local Storage. ■ The maximum number of characters for Syslog tabular alignment is 11.
Destination Host Before Manipulation [815]	Displays the original destination hostname (before manipulation, if any). **Note:** ■ The field is optional. You can include it in the CDR by CDR customization using the SBC CDR Format table.

Field	Description
	■ The field is applicable only to SBC signaling CDRs (all CDR Report Types). ■ The maximum number of characters for Syslog tabular alignment is 20.
Destination Host [813]	Displays the destination hostname (after manipulation, if any). **Note:** ■ The field is optional. You can include it in the CDR by CDR customization using the SBC CDR Format table. ■ The field is applicable only to SBC CDRs (all CDR Report Types). ■ The maximum number of characters for Syslog tabular alignment is 20.
Destination IP [403]	Displays the destination IP address. The field is a string of up to 20 characters. **Note:** ■ By default, the field is included in the CDR. ■ The field is applicable only to SBC signaling and Gateway CDRs (all CDR Report Types). ■ The default field title is "DestIp". ■ The maximum number of characters for Syslog tabular alignment is 20.
Destination Tags [441]	Displays destination tags. **Note:** ■ The field is optional. You can include it in the CDR by CDR customization using the SBC CDR Format and Gateway CDR Format tables. ■ The field is applicable only to SBC signaling and Gateway CDRs (all CDR Report Types). ■ The maximum number of characters for Syslog tabular alignment is 32.
Destination URI Before Manipulation [803]	Displays the original destination URI (username@host) before manipulation, if any. The field is a string of up to 150 characters. **Note:** ■ By default, the field is included in the CDR.

Field	Description
	■ The field is applicable only to SBC signaling CDRs (all CDR Report Types). ■ The default field title is "DstURIBeforeMap". ■ The maximum number of characters for Syslog tabular alignment is 41.
Destination URI [801]	Displays the destination URI (username@host) after manipulation, if any. The field is a string of up to 150 characters. **Note:** ■ By default, the field is included in the CDR. ■ The field is applicable only to SBC signaling CDRs (all CDR Report Types). ■ The default field title is "DstURI". ■ The maximum number of characters for Syslog tabular alignment is 41.
Destination Username Before Manipulation [811]	Displays the original destination username (before manipulation, if any). **Note:** ■ The field is optional. You can include it in the CDR by CDR customization using the SBC CDR Format tables. ■ The field is applicable only to SBC signaling CDRs (all CDR Report Types). ■ The maximum number of characters for Syslog tabular alignment is 20.
Destination Username [809]	Displays the destination username (after manipulation, if any). **Note:** ■ The field is optional. You can include it in the CDR by CDR customization using the SBC CDR Format tables. ■ The field is applicable only to SBC signaling CDRs (all CDR Report Types). ■ The maximum number of characters for Syslog tabular alignment is 20.
Direct Media [807]	Displays whether the call session flowed directly between the endpoints (i.e., Direct Media). The field is a string:

Field	Description
	■ "yes": The call is a direct media call session. ■ "no": The call traversed the device. **Note:** ■ By default, the field is included in the CDR. ■ The field is applicable only to SBC signaling CDRs (all CDR Report Types). ■ The default field title is "DirectMedia". ■ The maximum number of characters for Syslog tabular alignment is 11.
Endpoint Type [400]	Displays the endpoint type. The field is a string: ■ "NONE" ■ "SBC" (SBC calls) ■ "TEST" (for Test Call calls) ■ "3WCONF" (three-way conferencing calls) ■ "SIPREC" (SIPRec calls) ■ "MOH" (Music-on-Hold calls) **Note:** ■ By default, the field is included in the CDR. ■ The field is applicable only to SBC signaling and Gateway CDRs (all CDR Report Types). ■ The default field title is "EPTyp". ■ The maximum number of characters for Syslog tabular alignment is 10.
Global Session ID [309]	Displays the global session ID. **Note:** ■ The field is optional. You can include it in the CDR by CDR customization using the SBC CDR Format and Gateway CDR Format tables. ■ The field is applicable to SBC signaling and media, and Gateway CDRs. ■ The default field title is "h323-gw-id=" for RADIUS (A_ACCT_ SESSION_TIME).

Field	Description
	■ The maximum number of characters for Syslog tabular alignment is 16. ■ For more information on the global session ID, see Enabling Same Call Session ID over Multiple Devices on page 1099.
H323 ID [306]	Displays the device ID which can configured by the H323IDString parameter. It is typically used for RADIUS CDRs. The field is a string. **Note:** ■ The field is included in the default RADIUS CDR. ■ The field is applicable only to RADIUS SBC and Gateway CDRs (all CDR Report Types). ■ The default field title is "h323-gw-id for RADIUS. ■ The maximum number of characters for Syslog tabular alignment is 33.
IP Group ID [416]	Displays the IP Group ID. The field is an integer. **Note:** ■ The field is optional. You can include it in the CDR by CDR customization using the SBC CDR Format and Gateway CDR Format tables. ■ The field is applicable only to SBC signaling and Gateway CDRs (all CDR Report Types). ■ The maximum number of characters for Syslog tabular alignment is 5.
IP Group Name [417]	Displays the IP Group name. The field is a string of up to 40 characters. **Note:** ■ By default, the field is included in the CDR. ■ The field is applicable only to SBC signaling and Gateway CDRs (all CDR Report Types). ■ The default field title is "IPG (name)" for Gateway Syslog and Local Storage, "IPGroup (name)" for SBC Syslog and Local Storage, and "IP Group" in the Web SBC CDR History table. ■ The maximum number of characters for Syslog tabular alignment is 32.
IP Profile ID [425]	Displays the IP Profile ID. The field is an integer. **Note:**

Field	Description
	■ The field is optional. You can include it in the CDR by CDR customization using the SBC CDR Format and Gateway CDR Format tables. ■ The field is applicable only to SBC signaling and Gateway CDRs (all CDR Report Types). ■ The maximum number of characters for Syslog tabular alignment is 5.
IP Profile Name [426]	Displays the IP Profile name. The field is a string of up to 40 characters. **Note:** ■ By default, the field is included in the CDR. ■ The field is applicable only to SBC signaling and Gateway CDRs (all CDR Report Types). ■ The default field title is "IpProfileId (name)". ■ The maximum number of characters for Syslog tabular alignment is 32.
Is Recorded [822]	Displays if the SBC leg was recorded (SIPRec) or not. The field is a string: ■ "yes" ■ "no" **Note:** ■ The field is optional. You can include it in the CDR by CDR customization using the SBC CDR Format tables. ■ The field is applicable only to SBC signaling CDRs ("CALL_END" CDR Report Type; other Report Types will display "no"). ■ The maximum number of characters for Syslog tabular alignment is 5.
Latched RTP IP [631]	Displays the remote IP address of the incoming RTP stream that the device "latched" onto as a result of the RTP latching mechanism for NAT traversal. **Note:** ■ By default, the field is included in the CDR. ■ The field is applicable only to SBC media CDRs ("MEDIA_END" CDR Report Type) and Gateway CDRs ("CALL_END" CDR Report Type).

Field	Description
	■ The default field title is "LatchedRtpIp". ■ The maximum number of characters for Syslog tabular alignment is 20.
Latched RTP Port [632]	Displays the remote RTP port of the incoming RTP stream that the device "latched" onto as a result of the RTP latching mechanism for NAT traversal. The field is an integer 0 to 0xFFFF. **Note:** ■ By default, the field is included in the CDR. ■ The field is applicable only to SBC media CDRs ("MEDIA_END" CDR Report Type) and Gateway CDRs ("CALL_END" CDR Report Type). ■ The default field title is "LatchedRtpPort". ■ The maximum number of characters for Syslog tabular alignment is 15.
Latched T38 IP [633]	Displays the latching of a new T.38 stream (new IP address). **Note:** ■ By default, the field is included in the CDR. ■ The field is applicable only to SBC media CDRs ("MEDIA_END" CDR Report Type) and Gateway CDRs ("CALL_END" CDR Report Type). ■ The default field title is "LatchedT38Ip". ■ The maximum number of characters for Syslog tabular alignment is 20.
Latched T38 Port [634]	Displays the latching of a new T.38 stream (new port). The field is an integer 0 to 0xFFFF. **Note:** ■ By default, the field is included in the CDR. ■ The field is applicable only to SBC CDRs ("MEDIA_END" CDR Report Type) and Gateway CDRs ("CALL_END" CDR Report Type). ■ The default field title is "LatchedT38Port". ■ The maximum number of characters for Syslog tabular alignment is 15.
Leg ID [310]	Displays the unique ID of the call leg within a specific call session. The field is an integer. A basic SBC call consists of two legs (incoming and outgoing) and thus,

Field	Description
	two leg IDs are generated for the session, one for each leg. A basic Gateway call consists of only one leg ID. For each new call, the device assigns leg ID "1" to the first leg. The device then increments the leg ID for subsequent legs according to the leg sequence in the call session. For example, the device generates leg ID "1" for the SBC incoming leg and leg ID "2" for the SBC outgoing leg. If the call is transferred, the device generates leg ID "3" for the leg belonging to the call transfer target. Another example is a call forking session where the leg ID sequence may be as follows: incoming leg is "1", outgoing leg to user's office phone is "2" and outgoing leg to the user's mobile phone is "3". If the call is then transferred, the leg ID for the transfer leg is "4". **Note:** ■ By default, the field is included in the CDR. ■ The field is applicable only to SBC signaling and media, and Gateway CDRs ("CALL_START", "CALL_CONNECT" and "CALL_END" CDR Report Types). ■ The default field title is "LegId". ■ The maximum number of characters for Syslog tabular alignment is 5.
Local Input Octets [606]	Displays the local input octets (bytes). **Note:** ■ The field is optional. You can include it in the CDR by CDR customization using the Gateway CDR Format and SBC CDR Format tables. ■ The field is applicable only to SBC media CDRs ("MEDIA_END" CDR Report Type) and Gateway CDRs ("CALL_END" CDR Report Type). ■ The default field title is empty for RADIUS (ACCT_INPUT_OCTETS standard ID 42). ■ The maximum number of characters for Syslog tabular alignment is 10.
Local Input Packets [604]	Displays the number of packets received by the device. The field is an integer from 0 to 0XFFFFFFFF. **Note:** ■ By default, the field is included in the CDR. ■ The field is applicable only to SBC media CDRs ("MEDIA_END" CDR

Field	Description
	Report Type) and Gateway CDRs ("CALL_END" CDR Report Type). ■ The default field title is "InPackets" for Syslog and Local Storage, and empty for RADIUS (ACCT_INPUT_PACKETS). ■ The maximum number of characters for Syslog tabular alignment is 10.
Local Jitter [610]	Displays the RTP jitter. The field is an integer from 0 to 40000 samples (-1 if unavailable). **Note:** ■ By default, the field is included in the CDR. ■ The field is applicable only to SBC media CDRs ("MEDIA_END" CDR Report Type) and Gateway CDRs ("CALL_END" CDR Report Type). ■ The default field title is "RTPjitter". ■ The maximum number of characters for Syslog tabular alignment is 9.
Local MOS CQ [627]	Displays the local MOS for conversation quality. The field is an integer from 10 to 46 (127 if information is unavailable). **Note:** ■ By default, the field is included in the CDR. ■ The field is applicable only to SBC media CDRs ("MEDIA_END" CDR Report Type) and Gateway CDRs ("CALL_END" CDR Report Type). ■ The default field title is "LocalMosCQ". ■ The maximum number of characters for Syslog tabular alignment is 10.
Local Output Octets [607]	Displays the local output octets (bytes). **Note:** ■ The field is optional. You can include it in the CDR by CDR customization using the Gateway CDR Format and SBC CDR Format tables. ■ The field is applicable only to SBC media CDRs ("MEDIA_END" CDR Report Type) and Gateway CDRs ("CALL_END" CDR Report Type). ■ The default field title is empty for RADIUS (ACCT_OUTPUT_ OCTETS standard ID 43). ■ The maximum number of characters for Syslog tabular alignment is 10.

Field	Description
Local Output Packets [605]	Displays the number of packets sent by the device. The field is an integer from 0 to 0XFFFFFFFF. **Note:** ■ By default, the field is included in the CDR. ■ The field is applicable only to SBC media CDRs ("MEDIA_END" CDR Report Type) and Gateway CDRs ("CALL_END" CDR Report Type). ■ The default field title is "OutPackets" for Syslog and Local Storage, and empty for RADIUS (ACCT_OUTPUT_PACKETS standard ID 48). ■ The maximum number of characters for Syslog tabular alignment is 10.
Local Packet Loss [608]	Displays the number of packets lost of the entire stream. The field is an integer from 0 to 0xFFFFFFFF (-1 if information is unavailable). **Note:** ■ By default, the field is included in the CDR. ■ The field is applicable only to SBC media CDRs ("MEDIA_END" CDR Report Type) and Gateway CDRs ("CALL_END" CDR Report Type). ■ The default field title is "PackLoss" for Gateway Syslog and Local Storage, and "LocalPackLoss" for SBC Syslog. ■ The maximum number of characters for Syslog tabular alignment is 10.
Local R Factor [625]	Displays the local R-factor conversation quality. The field is an integer from 0 to 120 (127 if information is unavailable). **Note:** ■ By default, the field is included in the CDR. ■ The field is applicable only to SBC media CDRs ("MEDIA_END" CDR Report Type) and Gateway CDRs ("CALL_END" CDR Report Type). ■ The default field title is "LocalRFactor". ■ If the RTCP XR feature is unavailable (not licensed or disabled), this R-factor VoIP metric is not provided. Instead, the device sends the CDR field with the value 127, meaning that information is unavailable. ■ The maximum number of characters for Syslog tabular alignment is 12.
Local Round	Displays the average round-trip delay time of the entire RTP stream.

Field	Description
Trip Delay [609]	The field is an integer from 0 to 10000 ms (-1 if information is unavailable). **Note:** ■ By default, the field is included in the CDR. ■ The field is applicable only to SBC media CDRs ("MEDIA_END" CDR Report Type) and Gateway CDRs ("CALL_END" CDR Report Type). ■ The default field title is "RTPdelay". ■ The maximum number of characters for Syslog tabular alignment is 9.
Local RTP IP [620]	Displays the local RTP IP address. **Note:** ■ By default, the field is included in the CDR. ■ The field is applicable only to SBC media CDRs ("MEDIA_START" and "MEDIA_END" CDR Report Types) and Gateway CDRs ("CALL_ CONNECT" and "CALL_END" CDR Report Types). ■ The default field title is "LocalRtpIp". ■ The maximum number of characters for Syslog tabular alignment is 20.
Local RTP Port [621]	Displays the local RTP port. This field is an integer from 0 to 0xFFFF. **Note:** ■ By default, the field is included in the CDR. ■ The field is applicable only to SBC media CDRs ("MEDIA_START" and "MEDIA_END" CDR Report Types) and Gateway CDRs ("CALL_ CONNECT" and "CALL_END" CDR Report Types). ■ The default field title is "LocalRtpPort". ■ The maximum number of characters for Syslog tabular alignment is 15.
Local SSRC Sender [611]	Displays the local RTP synchronization source (SSRC). The field is an integer from 0 to 0XFFFFFFFF. **Note:** ■ By default, the field is included in the CDR. ■ The field is applicable only to SBC media CDRs ("MEDIA_END" CDR Report Type) and Gateway CDRs ("CALL_END" CDR Report Type) .

Field	Description
	■ The default field title is "RTPssrc" for Gateway Syslog and Local Storage, and "TxRTPssrc" for SBC Syslog. ■ The maximum number of characters for Syslog tabular alignment is 14.
MC Name [819]	Displays the Media Component name. The field is a string. **Note:** ■ The field is optional. You can include it in the CDR by CDR customization using the SBC CDR Format tables. ■ The field is applicable only to SBC signaling CDRs (all CDR Report Types). ■ The maximum number of characters for Syslog tabular alignment is 32.
Media List [819]	Displays all the media types (e.g., "audio", "text", "msrp", and "video") that was used for the call session. The field is a string. **Note:** ■ The field is optional. You can include it in the CDR by CDR customization using the SBC CDR Format table. ■ The field is applicable only to SBC signaling CDRs ("CALL_END" CDR Report Type). ■ The maximum number of characters for Syslog tabular alignment is 40.
Media Realm ID [427]	Displays the Media Realm ID. The field is an integer. **Note:** ■ The field is optional. You can include it in the CDR by CDR customization using the Gateway CDR Format and SBC CDR Format tables. ■ The field is applicable only to SBC signaling and Gateway CDRs (all CDR Report Types). ■ The maximum number of characters for Syslog tabular alignment is 5.
Media Realm Name [428]	Displays the Media Realm name. The field is a string of up to 40 characters. **Note:** ■ By default, the field is included in the CDR.

Field	Description
	■ The field is applicable only to SBC signaling and Gateway CDRs (all CDR Report Types). ■ The default field title is "MediaRealmId (name)". ■ The maximum number of characters for Syslog tabular alignment is 32.
Media Type [304]	Displays the media type (e.g., "audio", "text", or "video"). **Note:** ■ By default, the field is included in the CDR. ■ The field is applicable only to SBC media and Gateway CDRs ("CALL_END" and "MEDIA_END" CDR Report Type). ■ The default field title is "MediaType". ■ The maximum number of characters for Syslog tabular alignment is 10.
Packet Interval [602]	Displays the coder packet interval. The field is an integer from 10 to 200 ms. **Note:** ■ By default, the field is included in the CDR. ■ The field is applicable only to SBC media CDRS ("MEDIA_START", "MEDIA_UPDATE" and "MEDIA_END" CDR Report Types) and Gateway CDRs ("CALL_CONNECT" and "CALL_END" CDR Report Types). ■ The default field title is "Intrv". ■ The maximum number of characters for Syslog tabular alignment is 5.
Payload Type [603]	Displays the RTP payload type. The field is an integer, for example: ■ "0" for G.711 U-law ■ "8" for G.711 A-law ■ "18" for G.729 **Note:** ■ The field is optional. You can include it in the CDR by CDR customization using the SBC CDR Format and Gateway CDR Format tables. ■ The field is applicable only to SBC media CDRS ("MEDIA_START",

Field	Description
	"MEDIA_UPDATE" and "MEDIA_END" CDR Report Types) and Gateway CDRs ("CALL_CONNECT" and "CALL_END" CDR Report Types). ■ The maximum number of characters for Syslog tabular alignment is 5.
Proxy Set ID [424]	Displays the Proxy Set ID. **Note:** ■ The field is optional. You can include it in the CDR by CDR customization using the SBC CDR Format and Gateway CDR Format tables. ■ The field is applicable only to SBC signaling and Gateway CDRs (all CDR Report Types). ■ The maximum number of characters for Syslog tabular alignment is 10.
Proxy Set Name [438]	Displays the Proxy Set name. The field is a string of up to 40 characters. **Note:** ■ By default, the field is included in the CDR. ■ The field is applicable only to SBC signaling and Gateway CDRs (all CDR Report Types). ■ The default field title is "ProxySetId (name)". ■ The maximum number of characters for Syslog tabular alignment is 32.
RADIUS Call ID [307]	Displays the RADIUS call ID. **Note:** ■ The field is optional. You can include it in the CDR by CDR customization using the SBC CDR Format and Gateway CDR Format tables. ■ The field is applicable only to SBC and Gateway RADIUS CDRs (all CDR Report Types). ■ The default field title is "h323-conf-id=" in RADIUS CDRs. ■ The maximum number of characters for Syslog tabular alignment is 50.
Redirect Reason [414]	Displays the reason for the call redirection. The field is an integer of up to 15 digits:

Field	Description
	■ "-1": Not relevant ■ "0": Unknown reason ■ "1": Call forward busy (CFB) ■ "2": Call forward no reply (CFNR) ■ "3": Call forward network busy ■ "4": Call deflection ■ "5": Immediate call deflection ■ "6": Mobile subscriber not reachable ■ "9": DTE out of order ■ "10": Call forwarding DTE ■ "13": Call transfer ■ "14": Call pickup ■ "15": Call systematic or call forward unconditional (CFU) **Note:** ■ By default, the field is included in the CDR. ■ The field is applicable only to SBC signaling and Gateway CDRs ("CALL_END" CDR Report Types). ■ The default field title is "RedirectReason". ■ The maximum number of characters for Syslog tabular alignment is 15.
Redirect URI Before Manipulation [805]	Displays the original call redirect URI (username@host) before manipulation, if any. The field is a string of up to 150 characters. **Note:** ■ By default, the field is included in the CDR. ■ The field is applicable only to SBC signaling CDRs ("CALL_END" CDR Report Types). ■ The default field title is "RedirectURINumBeforeMap". ■ The maximum number of characters for Syslog tabular alignment is 41.
Redirect URI [804]	Displays the original call redirect URI (username@host) after manipulation, if any. The field value is a string of up to 150 characters.

Field	Description
	Note: ■ By default, the field is included in the CDR. ■ The field is applicable only to SBC signaling CDRs ("CALL_END" CDR Report Types). ■ The default field title is "RedirectURINum". ■ The maximum number of characters for Syslog tabular alignment is 41.
Release Time [413]	Displays the date and time the call ended (disconnected). The field is a string of up to 35 characters and presented in the following format: <hh:mm:ss:ms> UTC <DDD> <MMM> <DD> <YYYY>. For example, "17:00:55.002 UTC Thu Dec 14 2017". **Note:** ■ To configure the time zone string (e.g., "UTC" - default, "GMT+1" and "EST"), use the TimeZoneFormat parameter. ■ By default, the field is included in the CDR. ■ The field is applicable only to SBC signaling and Gateway CDRs ("CALL_END" CDR Report Types). ■ The default field title is "ReleaseTime" for Syslog, "h323-disconnect-time=" for RADIUS, and "Call End Time" in the Web SBC CDR History and Web Gateway CDR History tables. ■ The maximum number of characters for Syslog tabular alignment is 35.
Remote Input Octets [614]	Displays the remote input octets (bytes). **Note:** ■ The field is optional. You can include it in the CDR by CDR customization using the Gateway CDR Format and SBC CDR Format tables. ■ The field is applicable only to SBC media CDRs ("MEDIA_END" CDR Report Type) and Gateway CDRs ("CALL_END" CDR Report Type). ■ The maximum number of characters for Syslog tabular alignment is 10.
Remote Input Packets [612]	Displays the number of packets that the remote side reported it received. The field is an integer from 0 to 0XFFFFFFFF. **Note:**

Field	Description
	■ The field is optional. You can include it in the CDR by CDR customization using the Gateway CDR Format and SBC CDR Format tables. ■ The field is applicable only to SBC media CDRs ("MEDIA_END" CDR Report Type) and Gateway CDRs ("CALL_END" CDR Report Type). ■ The maximum number of characters for Syslog tabular alignment is 10.
Remote IP [404]	Displays the remote SIP IP address. **Note:** ■ The field is optional. You can include it in the CDR by CDR customization using the Gateway CDR Format and SBC CDR Format tables. ■ The field is applicable only to SBC signaling and Gateway CDRs ("CALL_START", "CALL_CONNECT", and "CALL_END" CDR Report Types). ■ The field is applicable to Syslog, RADIUS, Local Storage, and Web History CDRs. ■ The default CDR title is "Remote IP" in the Web SBC CDR History and Web Gateway CDR History tables. ■ The maximum number of characters for Syslog tabular alignment is 20.
Remote Jitter [618]	Displays the remote RTP jitter. The field is an integer from 0 to 40000 samples (-1 if unavailable). **Note:** ■ The field is optional. You can include it in the CDR by CDR customization using the Gateway CDR Format and SBC CDR Format tables. ■ The field is applicable only to SBC media CDRs ("MEDIA_END" CDR Report Type) and Gateway CDRs ("CALL_END" CDR Report Type). ■ The maximum number of characters for Syslog tabular alignment is 9.
Remote MOS CQ [628]	Displays the remote MOS for conversation quality. The field is an integer from 10 to 46 (127 if information is unavailable). **Note:** ■ By default, the field is included in the CDR.

Field	Description
	■ The field is applicable only to SBC media CDRs ("MEDIA_END" CDR Report Type) and Gateway CDRs ("CALL_END" CDR Report Type). ■ The default field title is "RemoteMosCQ". ■ The maximum number of characters for Syslog tabular alignment is 11.
Remote Output Octets [615]	Displays the remote output octets (bytes). **Note:** ■ The field is optional. You can include it in the CDR by CDR customization using the Gateway CDR Format and SBC CDR Format table. ■ The field is applicable only to SBC media CDRs ("MEDIA_END" CDR Report Type) and Gateway CDRs ("CALL_END" CDR Report Type). ■ The maximum number of characters for Syslog tabular alignment is 10.
Remote Output Packets [613]	Displays the number of packets received by the device. The field is an integer from 0 to 0XFFFFFFFF. **Note:** ■ The field is optional. You can include it in the CDR by CDR customization using the Gateway CDR Format and SBC CDR Format table. ■ The field is applicable only to SBC media CDRs ("MEDIA_END" CDR Report Type) and Gateway CDRs ("CALL_END" CDR Report Type). ■ The maximum number of characters for Syslog tabular alignment is 10.
Remote Packet Loss [616]	Displays the number of packets lost of the entire remote stream. The field is an integer from 0 to 0xFFFFFFFF (-1 if information is unavailable). **Note:** ■ By default, the field is included in the CDR. ■ The field is applicable only to SBC media CDRs ("MEDIA_END" CDR Report Type) and Gateway CDRs ("CALL_END" CDR Report Type). ■ The default field title is "RemotePackLoss". ■ The maximum number of characters for Syslog tabular alignment is 14.
Remote Port	Displays the remote SIP port. This field is an integer from 0 to 0xFFFF.

Field	Description
[407]	**Note:** ■ The field is optional. You can include it in the CDR by CDR customization using the Gateway CDR Format and SBC CDR Format table. ■ The field is applicable only to SBC signaling and Gateway CDRs ("CALL_START", "CALL_CONNECT", and "CALL_END" CDR Report Types). ■ The maximum number of characters for Syslog tabular alignment is 5.
Remote R Factor [626]	Displays the remote R-factor conversation quality. The field is an integer from 0 to 120 (127 if information is unavailable). **Note:** ■ By default, the field is included in the CDR. ■ The field is applicable only to SBC media CDRs ("MEDIA_END" CDR Report Type) and Gateway CDRs ("CALL_END" CDR Report Type). ■ The default field title is "RemoteRFactor". ■ If the RTCP XR feature is unavailable (not licensed or disabled), this R-factor VoIP metric is not provided. Instead, the device sends the CDR field with the value 127, meaning that information is unavailable. ■ The maximum number of characters for Syslog tabular alignment is 13.
Remote Round Trip Delay [617]	Displays the average round-trip delay time of the remote RTP stream. The field is an integer from 0 to 10000 ms (-1 if information is unavailable). **Note:** ■ The field is optional. You can include it in the CDR by CDR customization using the SBC CDR Format tables. ■ The field is applicable only to SBC media CDRs ("MEDIA_END" CDR Report Type) and Gateway CDRs ("CALL_END" CDR Report Type). ■ The maximum number of characters for Syslog tabular alignment is 9.
Remote RTP IP [622]	Displays the remote RTP IP address. **Note:** ■ By default, the field is included in the CDR.

Field	Description
	■ The field is applicable only to SBC media CDRs ("MEDIA_START", "MEDIA_UPDATE" and "MEDIA_END" CDR Report Types) and Gateway CDRs ("CALL CONNECT" and "CALL_END" CDR Report Types). ■ The default field title is "RtpIp" for Syslog Signaling and Local Storage, "RemoteRtpIp" for Syslog Media, and "h323-remote-address=" for RADIUS. ■ The maximum number of characters for Syslog tabular alignment is 20.
Remote RTP Port [623]	Displays the remote RTP port. This field is an integer from 0 to 0xFFFF. **Note:** ■ By default, the field is included in the CDR. ■ The field is applicable only to SBC media CDRs ("MEDIA_START", "MEDIA_UPDATE" and "MEDIA_END" CDR Report Types) and Gateway CDRs ("CALL CONNECT" and "CALL_END" CDR Report Types). ■ The default field title is ""Port" for Syslog Signaling and "RemoteRtpPort" for Syslog Media. ■ The maximum number of characters for Syslog tabular alignment is 5.
Remote SIP User Agent [818]	Displays the remote SIP User-Agent header value. **Note:** ■ The field is optional. You can include it in the CDR by CDR customization using the SBC CDR Format table. ■ The field is applicable only to SBC signaling ("CALL_START", "CALL_CONNECT" and "CALL_END" CDR Report Types). ■ The maximum number of characters for Syslog tabular alignment is 41.
Remote SSRC Sender [619]	Displays the remote (sender) RTP synchronization source (SSRC). The field is an integer from 0 to 0XFFFFFFFF. **Note:** ■ By default, the field is included in the CDR. ■ The field is applicable only to SBC media CDRs ("MEDIA_END" CDR Report Type) and Gateway CDRs ("CALL_END" CDR Report Type). ■ The default field title is "RemoteRTPssrc" for Gateway Syslog and

Field	Description
	"RxRTPssrc" for SBC Syslog Media. ■ The maximum number of characters for Syslog tabular alignment is 14.
Report Type [303]	Displays the type of CDR report. The field is a string: ■ "CALL_START": The CDR is sent upon an INVITE message. ■ "CALL_CONNECT": The CDR is sent upon a 200 OK response. ■ "CALL_END": The CDR is sent upon a BYE message. ■ "DIALOG_START": The CDR is sent upon the start of a non-INVITE session (only when enabled, using the EnableNonCallCdr parameter). ■ "DIALOG_END": The CDR is sent upon the end of a non-INVITE session (only when enabled, using the EnableNonCallCdr parameter). ■ "DIALOG_CONNECT ": The CDR is sent upon establishment of a non-INVITE session (only when enabled, using the EnableNonCallCdr parameter). ■ "MEDIA_START": The CDR is sent upon 200 OK response or early media ■ "MEDIA_UPDATE": The CDR is sent upon a re-INVITE message ■ "MEDIA_END": The CDR sent is upon a BYE message **Note:** ■ By default, the field is included in the CDR. ■ The field is applicable to SBC media and signaling, and Gateway CDRs. ■ The default field title is "GWReportType" for Gateway Syslog and Local Storage, "SBCReportType" for SBC Syslog and Local Storage, and "MediaReportType" for SBC Syslog Media. ■ The maximum number of characters for Syslog tabular alignment is 15.
RTP IP DiffServ [624]	The field displays the RTP IP DiffServ. The valid value is an integer from 0 to 63. **Note:** ■ By default, the field is included in the CDR.

Field	Description
	■ The field is applicable only to SBC media CDRs ("MEDIA_START", "MEDIA_UPDATE" and "MEDIA_END" CDR Report Types) and Gateway CDRs ("CALL CONNECT" and "CALL_END" CDR Report Types). ■ The default field title is "TxRTPIPDiffServ". ■ The maximum number of characters for Syslog tabular alignment is 15.
Session ID [302]	Displays the unique session ID. The field value is a string of up to 24 characters. **Note:** ■ By default, the field is included in the CDR. ■ The field is applicable to SBC media and signaling, and Gateway CDRs (all CDR Report Types). ■ The default field title is "SessionId". ■ The maximum number of characters for Syslog tabular alignment is 24.
Setup Time [411]	Displays the date and time that the call was setup. The field value is a string of up to 35 characters and presented in the following format: <hh:mm:ss:ms> UTC <DDD> <MMM> <DD> <YYYY>. For example, "17:00:49.052 UTC Thu Dec 14 2017" **Note:** ■ To configure the time zone string (e.g., "UTC" - default, "GMT+1", and "EST"), use the TimeZoneFormat parameter. ■ By default, the field is included in the CDR. ■ The field is applicable only to SBC signaling and Gateway CDRs (all CDR Report Types). ■ The default field title is "SetupTime"" for Syslog and Local Storage, and "h323-setup-time=" for RADIUS. ■ The maximum number of characters for Syslog tabular alignment is 35.
Signaling IP DiffServ [422]	Displays the signaling IP DiffServ. The field value is an integer of up to 15 digits. **Note:** ■ By default, the field is included in the CDR.

Field	Description
	■ The field is applicable only to SBC signaling and Gateway CDRs (all CDR Report Types). ■ The default field title is "TxSigIPDiffServ". ■ The maximum number of characters for Syslog tabular alignment is 15.
SIP Interface ID [420]	Displays the SIP Interface table row index (integer). **Note:** ■ The field is optional. You can include it in the CDR by CDR customization using the Gateway CDR Format and SBC CDR Format tables. ■ The field is applicable only to SBC signaling and Gateway CDRs (all CDR Report Types). ■ The maximum number of characters for Syslog tabular alignment is 5.
SIP Interface Name [433]	Displays the SIP Interface name. The field value is a string of up to 40 characters. **Note:** ■ By default, the field is included in the CDR. ■ The field is applicable only to SBC signaling and Gateway CDRs (all CDR Report Types). ■ The default field title is "SIPInterfaceId (name)". ■ The maximum number of characters for Syslog tabular alignment is 32.
SIP Local Tag [445]	Displays the 'tag' parameter of the SIP From / To headers that is generated by the device in the outgoing SIP message. The field value is a string of up to 100 characters. **Note:** ■ The field is optional. You can include it in the CDR by CDR customization using the Gateway CDR Format and SBC CDR Format tables. ■ The field is applicable to all CDR Report Types, but it may not be added to some Report Types in all call scenarios. ■ The field is applicable only to SBC signaling and Gateway CDRs. ■ The maximum number of characters for Syslog tabular alignment is

Field	Description
	20.
SIP Method [806]	Displays the SIP message type (method). The field value is a string of up to 10 characters: ■ "INVITE" ■ "OPTIONS" ■ "REGISTER" ■ "NOTIFY" ■ "INFO" ■ "SUBSCRIBE" ■ "MESSAGE" ■ "BENOTIFY" ■ "SERVICE" **Note:** ■ By default, the field is included in the CDR. ■ The field is applicable only to SBC signaling CDRs (all CDR Report Types). ■ The default field title is "SIPMethod". ■ The maximum number of characters for Syslog tabular alignment is 10.
SIP Remote Tag [446]	Displays the 'tag' parameter of the SIP From / To headers that is received by the device in the incoming SIP message. The field value is a string of up to 100 characters. **Note:** ■ The field is optional. You can include it in the CDR by CDR customization using the Gateway CDR Format and SBC CDR Format tables. ■ The field is applicable to all CDR Report Types, but it may not be added to some Report Types in all call scenarios. ■ The field is applicable only to SBC signaling and Gateway CDRs. ■ The maximum number of characters for Syslog tabular alignment is 20.
SIP Termination	Displays the description of the SIP call termination reason. The field

Field	Description
Description [430]	value is a string of up to 70 characters and is set to one of the following: ■ SIP Reason header, if exists, for example: SIP ;cause=200 ;text="Call completed elsewhere". ■ If no SIP Reason header exists, the description is taken from the reason text, if exists, of the SIP response code, for example: "417 Unknown Resource-Priority". ■ If no reason text exists in the SIP response code, the description is taken from an internal SIP response mapping mechanism. For example, if the device receives a SIP response "422", it sends in the CDR "422 Session Interval Too Small method" as the description. **Note:** ■ By default, the field is included in the CDR. ■ The field is applicable only to SBC signaling and Gateway CDRs ("CALL_END" CDR Report Types). ■ The default field title is "SipTermDesc". ■ The maximum number of characters for Syslog tabular alignment is 26.
SIP Termination Reason [429]	Displays the SIP reason for call termination. The field value is a string of up to 12 characters and is set to one of the following: ■ "BYE" ■ "CANCEL" ■ SIP error codes (e.g., "404") **Note:** ■ By default, the field is included in the CDR. ■ The field is applicable only to SBC signaling and Gateway CDRs ("CALL_END" CDR Report Types). ■ The default field title is "SIPTrmReason". ■ The maximum number of characters for Syslog tabular alignment is 12.
Source Host Before Manipulation [814]	Displays the original source hostname (before manipulation, if any). **Note:** ■ The field is optional. You can include it in the CDR by CDR customization using the SBC CDR Format table.

Field	Description
	■ The field is applicable only to SBC signaling CDRs (all CDR Report Types). ■ The maximum number of characters for Syslog tabular alignment is 20.
Source Host Name [517]	Displays the source hostname (after manipulation, if any). **Note:** ■ By default, the field is included in the CDR. ■ The field is applicable only to Gateway CDRs (all CDR Report Types). ■ The default field title is "SrcHost". ■ The maximum number of characters for Syslog tabular alignment is 20.
Source Host [812]	Displays the source hostname (after manipulation, if any). **Note:** ■ The field is optional. You can include it in the CDR by CDR customization using the SBC CDR Format table. ■ The field is applicable only to SBC signaling CDRs (all CDR Report Types). ■ The maximum number of characters for Syslog tabular alignment is 20.
Source IP [402]	Displays the source IP address. The field value is a string of up to 20 characters. **Note:** ■ By default, the field is included in the CDR. ■ The field is applicable only to SBC signaling and Gateway CDRs (all CDR Report Types). ■ The default field title is "SourceIp". ■ The maximum number of characters for Syslog tabular alignment is 20.
Source Port [405]	Displays the SIP signaling source UDP port. The field value is an integer of up to 10 digits. **Note:** ■ By default, the field is included in the CDR. ■ The field is applicable only to SBC signaling and Gateway CDRs (all

Field	Description
	CDR Report Types). ■ The default field title is "SigSourcePort" for Gateway Syslog, and "SourcePort" for SBC Syslog and Local Storage. ■ The maximum number of characters for Syslog tabular alignment is 13.
Source Tags [440]	Displays source tags. **Note:** ■ The field is optional. You can include it in the CDR by CDR customization using the SBC CDR Format and Gateway CDR Format tables. ■ The field is applicable only to SBC signaling and Gateway CDRs (all CDR Report Types). ■ The maximum number of characters for Syslog tabular alignment is 32.
Source URI Before Manipulation [802]	Displays the source URI (username@host) before manipulation. The field value is a string of up to 150 characters. **Note:** ■ By default, the field is included in the CDR. ■ The field is applicable only to SBC signaling CDRs (all CDR Report Types). ■ The default field title is "SrcURIBeforeMap". ■ The maximum number of characters for Syslog tabular alignment is 41.
Source URI [800]	Displays the source URI (username@host). The field value is a string of up to 150 characters. **Note:** ■ By default, the field is included in the CDR. ■ The field is applicable only to SBC signaling CDRs (all CDR Report Types). ■ The default field title is "SrcURI". ■ The maximum number of characters for Syslog tabular alignment is 41.
Source	Displays the original source username (before manipulation, if any).

Field	Description
Username Before Manipulation [810]	**Note:** ■ The field is optional. You can include it in the CDR by CDR customization using the SBC CDR Format tables. ■ The field is applicable only to SBC signaling CDRs (all CDR Report Types). ■ The default field title is "Caller" in the Web SBC CDR History table. ■ The maximum number of characters for Syslog tabular alignment is 20.
Source Username [808]	Displays the source username (after manipulation, if any). **Note:** ■ The field is optional. You can include it in the CDR by CDR customization using the SBC CDR Format tables. ■ The field is applicable only to SBC signaling CDRs (all CDR Report Types). ■ The maximum number of characters for Syslog tabular alignment is 20.
SRD ID [418]	Displays the SRD table row index. **Note:** ■ The field is optional. You can include it in the CDR by CDR customization using the Gateway CDR Format and SBC CDR Format tables. ■ The field is applicable only to SBC signaling and Gateway CDRs (all CDR Report Types). ■ The maximum number of characters for Syslog tabular alignment is 5.
SRD Name [419]	Displays the SRD name. The field value is a string of up to 40 characters. **Note:** ■ The field is optional. You can include it in the CDR by CDR customization using the Gateway CDR Format and SBC CDR Format tables. ■ The field is applicable only to SBC signaling and Gateway CDRs (all CDR Report Types). ■ The default field title is "SrdId (name)". ■ The maximum number of characters for Syslog tabular alignment is

Field	Description
	32.
Call Success [447]	Displays whether the call succeeded (“yes”) or failed (“no”). **Note:** ■ The field is optional. You can include it in the CDR by CDR customization using the SBC CDR Format and Gateway CDR Format tables. ■ The field is applicable only to SBC signaling and Gateway CDRs ("CALL_END" CDR Report Types).
Termination Reason Category [423]	Displays the category of the call termination reason. The field value is up to 17 characters and is set to one of the following: **Calls with duration 0 (i.e., not connected):** ■ "NO_ANSWER": ✓ "GWAPP_NORMAL_CALL_CLEAR" ✓ "GWAPP_NO_USER_RESPONDING" ✓ "GWAPP_NO_ANSWER_FROM_USER_ALERTED" ■ "BUSY": ✓ "GWAPP_USER_BUSY" ■ "NO_RESOURCES": ✓ "GWAPP_RESOUUCE_UNAVAILABLE_UNSPECIFIED" ✓ "RELEASE_BECAUSE_NO_CONFERENCE_RESOURCES_LEFT" ✓ "RESOURCE_BECAUSE_NO_TRANSCODING_RESOURCES_LEFT" ✓ "RELEASE_BECAUSE_GW_LOCKED" ■ "NO_MATCH": ✓ "RELEASE_BECAUSE_UNMATCHED_CAPABILITIES" ■ "FORWARDED": ✓ "RELEASE_BECAUSE_FORWARD" ■ "GENERAL_FAILED": Any other reason **Calls with duration:** ■ "NORMAL_CALL_CLEAR": ✓ "GWAPP_NORMAL_CALL_CLEAR" ■ "ABNORMALLY_TERMINATED": Anything else

Field	Description
	Note: ■ By default, the field is included in the CDR. ■ The field is applicable only to SBC signaling and Gateway CDRs ("CALL_END" CDR Report Types). ■ The default field title is "TrmReasonCategory" for Syslog and Local Storage, and "Termination Reason" for Web CDR History. ■ The maximum number of characters for Syslog tabular alignment is 17.
Termination Reason Value [437]	Displays the Q.850 reason codes (1-127) for call termination. For example, "16" for Normal Termination. **Note:** ■ By default, the field is included in the CDR for RADIUS CDRs. ■ The field is applicable only to SBC signaling and Gateway CDRs ("CALL_END" CDR Report Types). ■ The default field title is "h323-disconnect-cause=" (e.g., "h323-disconnect-cause=16"). ■ The maximum number of characters for Syslog tabular alignment is 5.
Termination Reason [410]	Displays the reason for the call termination. The field value is a string of up to 40 characters and is set to one of the following: ■ Standard Call Termination Reasons: ✓ "GWAPP_REASON_NOT_RELEVANT" (0) ✓ " GWAPP_ALL_RELEASE_REASONS" (0) ✓ "GWAPP_UNASSIGNED_NUMBER" (1) ✓ "GWAPP_NO_ROUTE_TO_TRANSIT_NET" (3) ✓ "GWAPP_NO_ROUTE_TO_DESTINATION" (3) ✓ "GWAPP_SEND_SPECIAL_INFORMATION_TONE" (4) ✓ "GWAPP_MISDIALED_TRUNK_PREFIX" (5) ✓ "GWAPP_CHANNEL_UNACCEPTABLE" (6) ✓ "GWAPP_CALL_AWARDED_AND" (7) ✓ "GWAPP_PREEMPTION" (8) ✓ "GWAPP_PREEMPTION_CIRCUIT_RESERVED_FOR_REUSE" (9)

Field	Description
	✓ "GWAPP_NORMAL_CALL_CLEAR" (16) ✓ "GWAPP_USER_BUSY" (17) ✓ "GWAPP_NO_USER_RESPONDING" (18) ✓ "GWAPP_NO_ANSWER_FROM_USER_ALERTED" (19) ✓ "MFCR2_ACCEPT_CALL" (20) ✓ "GWAPP_CALL_REJECTED" (21) ✓ "GWAPP_NUMBER_CHANGED" (22) ✓ "GWAPP_REDIRECTION" (23) ✓ "GWAPP_EXCHANGE_ROUTING_ERROR" (25) ✓ "GWAPP_NON_SELECTED_USER_CLEARING" (26) ✓ "GWAPP_INVALID_NUMBER_FORMAT" (28) ✓ "GWAPP_FACILITY_REJECT" (29) ✓ "GWAPP_RESPONSE_TO_STATUS_ENQUIRY" (30) ✓ "GWAPP_NORMAL_UNSPECIFIED" (31) ✓ "GWAPP_CIRCUIT_CONGESTION" (32) ✓ "GWAPP_USER_CONGESTION" (33) ✓ "GWAPP_NO_CIRCUIT_AVAILABLE" (34) ✓ "GWAPP_NETWORK_OUT_OF_ORDER" (38) ✓ "GWAPP_PERM_FR_MODE_CONN_OUT_OF_S" (39) ✓ "GWAPP_PERM_FR_MODE_CONN_OPERATIONAL" (40) ✓ "GWAPP_NETWORK_TEMPORARY_FAILURE" (41) ✓ "GWAPP_NETWORK_CONGESTION" (42) ✓ "GWAPP_ACCESS_INFORMATION_DISCARDED" (43) ✓ "GWAPP_REQUESTED_CIRCUIT_NOT_AVAILABLE" (44) ✓ "GWAPP_PRECEDENCE_CALL_BLOCKED" (46) ✓ "GWAPP_RESOURCE_UNAVAILABLE_UNSPECIFIED" (47) ✓ "GWAPP_QUALITY_OF_SERVICE_UNAVAILABLE" (49) ✓ "GWAPP_REQUESTED_FAC_NOT_SUBSCRIBED" (50) ✓ "GWAPP_BC_NOT_AUTHORIZED" (57) ✓ "GWAPP_BC_NOT_PRESENTLY_AVAILABLE" (58)

Field	Description
	✓ "GWAPP_SERVICE_NOT_AVAILABLE" (63) ✓ "GWAPP_CUG_OUT_CALLS_BARRED" (53) ✓ "GWAPP_CUG_INC_CALLS_BARRED" (55) ✓ "GWAPP_ACCES_INFO_SUBS_CLASS_INCONS" (62) ✓ "GWAPP_BC_NOT_IMPLEMENTED" (65) ✓ "GWAPP_CHANNEL_TYPE_NOT_IMPLEMENTED" (66) ✓ "GWAPP_REQUESTED_FAC_NOT_IMPLEMENTED" (69) ✓ "GWAPP_ONLY_RESTRICTED_INFO_BEARER" (70) ✓ "GWAPP_SERVICE_NOT_IMPLEMENTED_UNSPECIFIED" (79) ✓ "GWAPP_INVALID_CALL_REF" (81) ✓ "GWAPP_IDENTIFIED_CHANNEL_NOT_EXIST" (82) ✓ "GWAPP_SUSPENDED_CALL_BUT_CALL_ID_NOT_EXIST" (83) ✓ "GWAPP_CALL_ID_IN_USE" (84) ✓ "GWAPP_NO_CALL_SUSPENDED" (85) ✓ "GWAPP_CALL_HAVING_CALL_ID_CLEARED" (86) ✓ "GWAPP_INCOMPATIBLE_DESTINATION" (88) ✓ "GWAPP_INVALID_TRANSIT_NETWORK_SELECTION" (91) ✓ "GWAPP_INVALID_MESSAGE_UNSPECIFIED" (95) ✓ "GWAPP_NOT_CUG_MEMBER" (87) ✓ "GWAPP_CUG_NON_EXISTENT" (90) ✓ "GWAPP_MANDATORY_IE_MISSING" (96) ✓ "GWAPP_MESSAGE_TYPE_NON_EXISTENT" (97) ✓ "GWAPP_MESSAGE_STATE_INCONSISTENCY" (98) ✓ "GWAPP_NON_EXISTENT_IE" (99) ✓ "GWAPP_INVALID_IE_CONTENT" (100) ✓ "GWAPP_MESSAGE_NOT_COMPATIBLE" (101) ✓ "GWAPP_RECOVERY_ON_TIMER_EXPIRY" (102) ✓ "GWAPP_PARAMETER_NON_EXISTENT" (103) ✓ "GWAPP_MESSAGE_WITH_UNRECOGNIZED_PARAM" (110) ✓ "GWAPP_PROTOCOL_ERROR_UNSPECIFIED" (111)

Field	Description
	✓ "GWAPP_UKNOWN_ERROR" (112) ✓ "GWAPP_INTERWORKING_UNSPECIFIED" (127) ■ AudioCodes Proprietary: ✓ "RELEASE_BECAUSE_UNKNOWN_REASON" (304) ✓ "RELEASE_BECAUSE_TRUNK_DISCONNECTED" (305) ✓ "RELEASE_BECAUSE_REMOTE_CANCEL_CALL" (306) ✓ "RELEASE_BECAUSE_UNMATCHED_CAPABILITIES" (307) ✓ "RELEASE_BECAUSE_UNMATCHED_CREDENTIALS" (308) ✓ "RELEASE_BECAUSE_UNABLE_TO_HANDLE_REMOTE_REQUEST" (309) ✓ "RELEASE_BECAUSE_NO_CONFERENCE_RESOURCES_LEFT" (310) ✓ "RELEASE_BECAUSE_CONFERENCE_FULL" (311) ✓ "RELEASE_BECAUSE_MANUAL_DISC" (315) ✓ "RELEASE_BECAUSE_SILENCE_DISC" (316) ✓ "RELEASE_BECAUSE_NORTEL_XFER_SUCCESS" (317) ✓ "RELEASE_BECAUSE_RTP_CONN_BROKEN" (318) ✓ "RELEASE_BECAUSE_DISCONNECT_CODE" (319) ✓ "RELEASE_BECAUSE_GW_LOCKED" (320) ✓ "RELEASE_BECAUSE_FAIL" (321) ✓ "RELEASE_BECAUSE_FORWARD" (322) ✓ "RELEASE_BECAUSE_ANONYMOUS_SOURCE" (323) ✓ "PREEMPTION_ANALOG_CIRCUIT_RESERVED_FOR_REUSE" (324) ✓ "RELEASE_BECAUSE_PRECEDENCE_CALL_BLOCKED" (325) ✓ "RELEASE_BECAUSE_HELD_TIMEOUT" (326) ✓ "RELEASE_BECAUSE_MEDIA_MISMATCH" (327) ✓ "RELEASE_BECAUSE_MAX_DURATION_TIMER_EXPIRED" (328) ✓ "RELEASE_BECAUSE_TRANSCODING_FULL" (329) ✓ "RELEASE_BECAUSE_NO_TRANSCODING_RESOURCES_LEFT" (330)

Field	Description
	✓ "RELEASE_POSTPONE_POSSIBLE" (331) ✓ "RELEASE_BECAUSE_PREEMPTION_DUE_TO_HIGH_PRIORITY" (332) ✓ "RELEASE_BECAUSE_PREEMPTION_FAILED" (333) ✓ "RELEASE_BECAUSE_IP_PROFILE_CALL_LIMIT" (805) ✓ "RELEASE_BECAUSE_OUT_MEDIA_LIMITS_EXCEEDED" (806) ✓ "RELEASE_BECAUSE_CALL_TRANSFERRED" (807) ✓ "RELEASE_BECAUSE_CLASSIFICATION_FAILED" (808) ✓ "RELEASE_BECAUSE_AUTHENTICATION_FAILED" (809) ✓ "RELEASE_BECAUSE_ARM_DROP" (811) ✓ "RELEASE_BECAUSE_MEDIA_DEST_UNREACHABLE" (812) ✓ "RELEASE_BECAUSE_START_ARM_ROUTING" (813) ✓ "RELEASE_BECAUSE_FORWARD_SUPPLEMENTARY" (814) ✓ "RELEASE_BECAUSE_FAX_REROUTING" (815) ✓ "RELEASE_BECAUSE_LDAP_FAILURE" (816) ✓ "RELEASE_BECAUSE_CALLSETUPRULES_FAILURE" (817) ✓ "RELEASE_BECAUSE_NO_USER_FOUND" (818) ✓ "RELEASE_BECAUSE_IN_ADMISSION_FAILED" (819) ✓ "RELEASE_BECAUSE_OUT_ADMISSION_FAILED" (820) ✓ "RELEASE_BECAUSE_IN_MEDIA_LIMITS_EXCEEDED" (821) ✓ "RELEASE_BECAUSE_USER_BLOCKED" (822) ✓ "RELEASE_BECAUSE_BAD_INFO_PACKAGE" (823) ✓ "RELEASE_BECAUSE_SRC_IP_IS_NOT_DEDICATED_REGISTRAR" (824) ✓ "RELEASE_BECAUSE_ACD_THRESHOLD_CROSSED" (850) ✓ "RELEASE_BECAUSE_ASR_THRESHOLD_CROSSED" (851) ✓ "RELEASE_BECAUSE_NER_THRESHOLD_CROSSED" (852) ✓ "RELEASE_BECAUSE_IPGROUP_REGISTRATION_MODE" (853) ✓ "RELEASE_BECAUSE_FEATUREKEY_CHANGED" (854) ✓ "RELEASE_BECAUSE_INTERNAL_ROUTE" (855)

Field	Description
	✓ "RELEASE_BECAUSE_CID_CMD_FAILURE" (856) ✓ "RELEASE_BECAUSE_OTHER_FORKED_CALL_ANSWERED" (857) ✓ "RELEASE_BECAUSE_MEDIA_SYNC_FAILED" (858) ✓ "RELEASE_BECAUSE_REG_MAX_THRESHOLD_CROSSED" (859) ✓ "RELEASE_BECAUSE_PUSH_NOTIFICATION_FAILED" (860) **Note:** ■ By default, the field is included in the CDR. ■ The field is applicable only to SBC signaling and Gateway CDRs ("CALL_END" CDR Report Types). ■ The default field title is "TrmReason". ■ The maximum number of characters for Syslog tabular alignment is 40.
Termination Side RADIUS [435]	Displays the party that terminated the call. The field value is a string: ■ "originate": SBC incoming leg ■ "answer": SBC outgoing leg **Note:** ■ By default, the field is included in the CDR. ■ The field is mainly relevant to RADIUS CDRs, but can also be used in Syslog and Local Storage. ■ The default field title is "terminator=". ■ The maximum number of characters for Syslog tabular alignment is 10.
Termination Side Yes No [436]	Displays the party that terminated the call. The field value is a string: ■ "yes": SBC outgoing leg ■ "no": SBC incoming leg The field is applicable to RADIUS CDRs **Note:** ■ By default, the field is included in the CDR. ■ The field is mainly relevant to RADIUS CDRs, but can also be used in Syslog and Local Storage. ■ The default field title is "terminator=" (e.g., "terminator=yes").

Field	Description
	■ The maximum number of characters for Syslog tabular alignment is 5.
Termination Side [409]	Displays the party that terminated the call. The field value is a string: ■ "LCL": SBC Outgoing leg or Tel side. ■ "RMT": SBC Incoming leg or IP side. ■ "UNKN": Unknown For example, if the Orig field is "RMT" and this Termination Side field is "LCL", then the called party ended the call. **Note:** ■ By default, the field is included in the CDR. ■ The field is applicable only to SBC signaling and Gateway CDRs ("CALL_END" CDR Report Types). ■ The default field title is "TrmSd". ■ The maximum number of characters for Syslog tabular alignment is 5.
Transport Type [421]	Displays the SIP signaling transport type protocol. The field value is a string: ■ "UDP" ■ "TCP" ■ "TLS" **Note:** ■ By default, the field is included in the CDR. ■ The field is applicable only to SBC signaling and Gateway CDRs (all CDR Report Types). ■ The default field title is "SigTransportType" for Gateway Syslog and Local Storage, and "TransportType" for SBC Syslog and Local Storage. ■ The maximum number of characters for Syslog tabular alignment is 16.
Trigger [439]	Displays the reason for the call (i.e., what triggered it): ■ "Normal": regular call ■ "Refer": call transfer

Field	Description
	■ "AltRoute": alternative routing ■ "Forward": call forward ■ "Reroute": When a broken connection on the outgoing leg occurs, the call is rerouted to another destination according to the IP-to-IP Routing table (where matching characteristics includes the trigger for reroute). **Note:** ■ By default, the field is included in the CDR. ■ The field is applicable only to SBC signaling and Gateway CDRs (all CDR Report Types). ■ The default field title is "Trigger". ■ The maximum number of characters for Syslog tabular alignment is 8.
Trunk Group ID [503]	Displays the Trunk Group ID (integer). **Note:** ■ By default, the field is included in the CDR. ■ The field is applicable only to Gateway CDRs (all CDR Report Types). ■ The default field title is "TG". ■ The maximum number of characters for Syslog tabular alignment is 5.
Var Call User Defined 1-5 [448-452]	Displays the SIP header data obtained from call variables (Var.Call.Src/Dst.UserDefined1-5) in Message Manipulation rules. **Note:** ■ The field is optional. You can include it in the CDR by CDR customization using the SBC CDR Format table. ■ The field is applicable only to SBC signaling (all CDR Report Types). ■ The field is applicable only to Syslog, RADIUS, Local Storage, and JSON. ■ For each variable-based field, the maximum number of characters for Syslog tabular alignment is 20. ■ The maximum characters for all five variable-based CDR fields together is 200. For example, if the summation of Var Call User Defined 1 and Var Call User Defined 2 is 200 characters, no characters are displayed for the other variables.

Field	Description
Was Call Started [415]	Displays if the call was started or not (i.e., if a "CALL_START" CDR Report was generated). ■ "0": No INVITE was sent to the IP side for the Tel-to-IP call, or no Setup message was sent to the Tel side for the IP-to-Tel call. Note that the first "CALL_START" CDR report type of a new signaling leg has value "0". ■ "1": The call was started – an INVITE was sent to the IP side for the Tel-to-IP call, or a Setup message was sent to the Tel side for the IP-to-Tel call. **Note:** ■ The field is optional. You can include it in the CDR by CDR customization using the Gateway CDR Format and SBC CDR Format table. ■ The field is applicable only to SBC signaling and Gateway CDRs ("CALL_END" CDR Report Types). ■ The field is applicable only to Syslog, RADIUS, and Local Storage. ■ The maximum number of characters for Syslog tabular alignment is 5.
Coder Transcoding [635]	Displays whether there was coder transcoding for the SBC call. The field is a string: ■ "TRANSCODING" ■ "NO_TRANSCODING" **Note:** ■ The field is optional. You can include it in the CDR by CDR customization using the SBC CDR Format table. ■ The field is applicable only to SBC media CDRs ("MEDIA_END" CDR Report Types). ■ The field is applicable only to Syslog and RADIUS. ■ The maximum number of characters for Syslog tabular alignment is 17.
Voice AI Connector ID [820]	Displays the ID of the VoiceAI Connect when the device is used as a VoiceAI Connect. **Note:** ■ The field is optional. You can include it in the CDR by CDR

Field	Description
	customization using the SBC CDR Format table. ■ The field is applicable only to SBC signaling (all CDR Report Types). ■ The default field title is "VoiceAIConnectorName". ■ For more information on VoiceAI Connect, go to https://techdocs.audiocodes.com/voice-ai-connect.
Voice AI Connector Name [821]	Displays the name of the VoiceAI Connect when the device is used as a VoiceAI Connect. **Note:** ■ By default, the field is included in the CDR. ■ The field is applicable only to SBC signaling (all CDR Report Types). ■ The default field title is "VoiceAIConnectorName". ■ For more information on the VoiceAI Connect, go to https://techdocs.audiocodes.com/voice-ai-connect.

Customizing CDRs for SBC Calls and Test Calls

The SBC CDR Format table lets you customize CDRs for SBC calls and CDRs for Test Calls that are generated by the device for the following CDR types:

- CDRs for SIP signaling or media sent in Syslog messages. For CDRs sent in Syslog messages, you can customize the name of the CDR field. You can configure up to 128 Syslog CDR customization rules.
- CDRs for RADIUS accounting and sent in RADIUS accounting request messages. For RADIUS accounting CDRs, you can customize the RADIUS Attribute's prefix name and RADIUS Attribute's ID, for standard RADIUS Attributes and vendor-specific RADIUS Attributes (VSA). For example, instead of the default VSA name "h323-connect-time" with RADIUS Attribute ID 28, you can change the name to "Call-Connect-Time" with ID 29. You can configure up to 128 RADIUS-Accounting CDR customization rules (i.e., maximum number of RADIUS Attributes that the device can include in the CDR). For more information on RADIUS accounting, see Configuring RADIUS Accounting.
- CDRs stored locally on the device. For local storage of CDRs, you can customize the name of the CDR field. You can configure up to 64 locally-stored CDR customization rules. For more information on storing CDRs on the device, see Storing CDRs on the Device.
- CDRs (signaling only) sent to the REST server in JSON format using the device's REST API. You can configure up to 64 JSON CDR customization rules. For more information on CDRs and REST, see Configuring CDR Reporting to REST Server on page 986.

Customizing the CDR means the following:

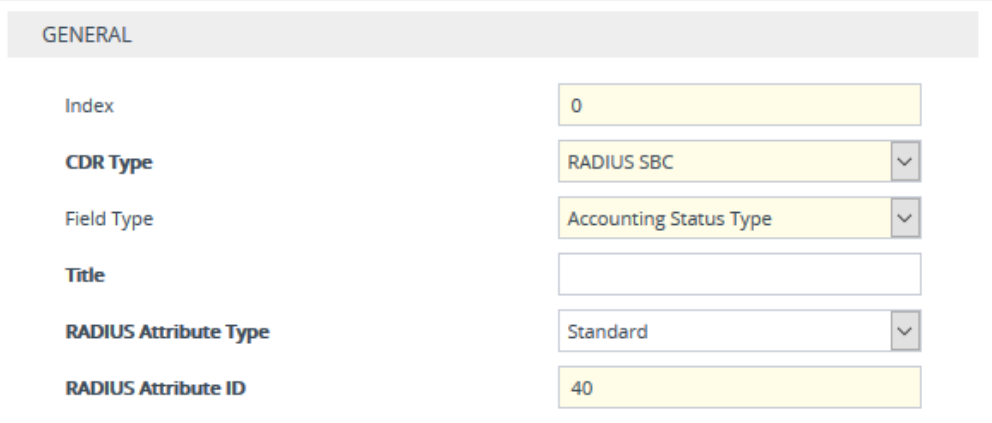
- Defining which CDR fields are included in the CDR. For example, if you configure only one customization rule for the Syslog signaling (SBC) CDR type with the Call Duration CDR field, the device generates these CDR types with only this single CDR field.

  You can also customize the CDR to include a user-defined CDR field based on any SIP header information. This is done by using Message Manipulation rules with the call variables `Var.Call.Src/Dst.<Variable Name>`, where `Variable Name` is UserDefined1, UserDefined2, UserDefined3, UserDefined4 or UserDefined5. The Message Manipulation rule stores the SIP header value in the variable. When you customize the CDR in the SBC CDR Format table, you need to select the same variable (**Var Call User Defined 1-5**) in the 'CDR Field Type' parameter that you used in the Message Manipulation rule. When the device generates the CDR, it retrieves the stored information from the variable and adds it to the CDR under your customized CDR field title. If a variable is not added or modified in the Message Manipulation rule, and the CDR is customized to include its stored value, the CDR displays an empty string for the value. For an example, see Example of Call Variables for CDR Customization on page 1043.

- Changing the default name (`title`) of the CDR field. For example, you can change the title of the Call Duration CDR field to "Call Length".

- Changing the RADIUS Attribute's prefix name and ID, for standard RADIUS Attributes and vendor-specific RADIUS Attributes (VSA).

- If you don't customize the CDR, the device generates the CDR in a default format (fields and titles). For a detailed description of the fields that can be included in the CDR (customized and default), see CDR Field Description.
- To return to the default CDR format for a specific CDR type, remove all the customization rules of that CDR type.
- When customizing the RADIUS CDR:
  - ✓ The following standard RADIUS Attributes cannot be customized: 1 through 6, 18 through 20, 22, 23, 27 through 29, 32, 34 through 39, 41, 44, 52, 53, 55, 60 through 85, 88, 90, and 91.
  - ✓ You must add the following RADIUS Attribute as the **first** rule in the SBC CDR Format table to ensure uniqueness (and to differentiate) between Call Connect (START) and Call End (STOP) RADIUS packets:

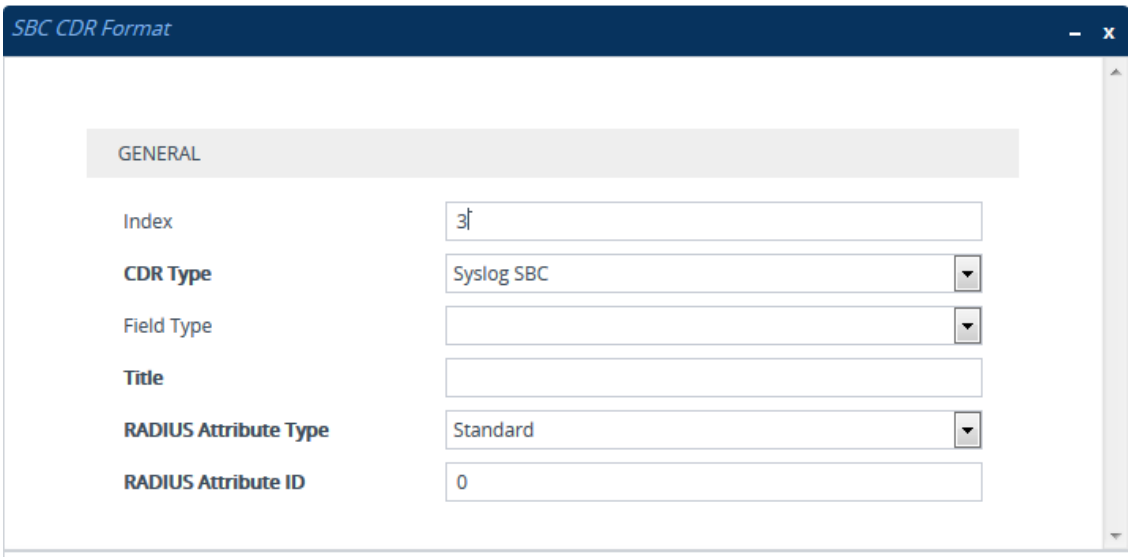


- If the RTCP XR feature is unavailable (not licensed or disabled), the R-factor VoIP metrics are not provided in CDRs (CDR fields, Local R Factor and Remote R Factor) generated by the device. Instead, these CDR fields are sent with the value 127, meaning that information is unavailable.
- Test Call CDRs include "CALL_START", "CALL_CONNECT" and "CALL_END" CDR Report Types.
- By default, SBC signaling CDRs that are sent at the end of the call ("CALL_END" CDR Report Type) include only signaling-related CDR fields. However, by using the SBC CDR Format table, you can customize this CDR to also include media-related fields.
- To view historical CDRs stored on the devicesee Viewing CDR History of SBC and Test Calls on page 970.

The following procedure describes how to customize SBC and Test Call CDRs through the Web interface. You can also configure it through ini file [SBCCDRFormat] or CLI (`configure troubleshoot > cdr > cdr-format sbc-cdr-format`).

➢ To customize SBC and Test Call CDRs:

1. Open the SBC CDR Format table (**Troubleshoot** menu > **Troubleshoot** tab > **Call Detail Record** folder > **SBC CDR Format**).
2. Click **New**; the following dialog box appears:



3. Configure the CDR according to the parameters described in the table below.
4. Click **Apply**.

Examples of configured CDR customization rules are shown below:

INDEX	CDR TYPE	FIELD TYPE	TITLE	RADIUS ATTRIBUTE TYPE	RADIUS ATTRIBUTE ID
0	Syslog SBC	Source IP	"Source IP Address"	Standard	0
1	RADIUS SBC	Release Time	disconnect-time=	Vendor Specific	29
2	Local Storage SBC	Call Duration	Lenght of Call	Standard	0

Table 50-3: SBC CDR Format Table Parameter Descriptions

Parameter	Description
'Index' [SBCCDRFormat_Index]	Defines an index number for the new table row. **Note:** Each row must be configured with a unique index.
'CDR Type' `cdr-type` [SBCCDRFormat_CDRType]	Defines the application type for which you want to customize CDRs. ■ [1] **Syslog SBC** = (Default) Customizes CDR fields for SIP signaling-related CDRs sent in Syslog messages. However, for SBC signaling "CALL_END" CDR Report Types (sent at the end of the call), you can also customize the CDR to include media-related CDR fields (e.g., Local Packet Loss). ■ [3] **Syslog Media** = Customizes CDR fields for media-related CDRs sent in Syslog messages. ■ [5] **Local Storage SBC** = Customizes CDR fields that are stored locally on the device. Only signaling-related CDRs are stored locally on the device. However, for SBC signaling "CALL_END" CDR Report Types (sent at the end of the call), you can also customize the CDR to include media-related CDR fields (e.g.,

Parameter	Description
	Local Packet Loss). ■ [7] **RADIUS SBC** = Customizes CDR fields (i.e., RADIUS Attributes) for CDRs sent in RADIUS accounting request messages. ■ [11] **JSON SBC** = Customizes CDR fields for SIP signaling-related CDRs that are sent in JSON format to the REST server using the device's REST API.
'Field Type' `col-type` [SBCCDRFormat_FieldType]	Defines the CDR field (column) that you want to customize. The applicable CDR field depends on the settings of the 'CDR Type' parameter: ■ For all types: [300] **CDR Type** (default); [301] **Call ID**; [302] **Session ID**; [303] **Report Type**; [304] **Media Type**; [305] **Accounting Status Type**; [306] **H323 ID**; [307] **RADIUS Call ID**; [308] **Blank**; [309] **Global Session ID**; [310] **Leg ID** ■ **Syslog SBC** (signaling), **Local Storage SBC**, **RADIUS SBC**, and **JSON SBC**: [400] **Endpoint Type**; [401] **Call Orig**; [402] **Source IP**; [403] **Destination IP**; [404] **Remote IP**; [405] **Source Port**; [406] **Dest Port**; [407] **Remote Port**; [408] **Call Duration**; [409] **Termination Side**; [410] **Termination Reason**; [411] **Setup Time**; [412] **Connect Time**; [413] **Release Time**; [414] **Redirect Reason**; [415] **Was Call Started**; [416] **IP Group ID**; [417] **IP Group Name**; [418] **SRD ID**; [419] **SRD Name**; [420] **SIP Interface ID**; [421] **Transport Type**; [422] **Signaling IP DiffServ**; [423] **Termination Reason Category**; [424] **Proxy Set ID**; [425] **IP Profile ID**; [426] **IP Profile Name**; [427] **Media Realm ID**; [428] **Media Realm Name**; [429] **SIP Termination Reason**; [430] **SIP Termination Description**; [431] **Caller Display ID**; [432] **Callee Display ID**; [433] **SIP Interface Name**; [434] **Call Orig RADIUS**; [435] **Termination Side RADIUS**; [436] **Termination Side Yes No**; [437] **Termination Reason Value;** [438] **Proxy Set Name**; [439] **Trigger**; [442] **Call End Sequence Number**; [443] **Alerting Time**; [445] **SIP Local Tag**; [446] **SIP Remote Tag**; [447] **Call Success**; [448] **Var Call User Defined 1**; [449] **Var Call User Defined 2**; [450] **Var Call User Defined 3**; [451] **Var Call User Defined 4**; [452] **Var Call User Defined 5** ■ **Syslog Media**, **RADIUS SBC**, **Local Storage SBC**, and **Syslog SBC**: [600] **Channel ID**; [601] **Coder Type**; [602] **Packet Interval**; [603] **Payload Type**; [604] **Local Input Packets**; [605] **Local Output**

Parameter	Description
	Packets; [606] **Local Input Octets**; [607] **Local Output Octets**; [608] **Local Packet Loss**; [609] **Local Round Trip Delay**; [610] **Local Jitter**; [611] **Local SSRC Sender**; [612] **Remote Input Packets**; [613] **Remote Output Packets**; [614] **Remote Input Octets**; [615] **Remote Output Octets**; [616] **Remote Packet Loss**; [617] **Remote Round Trip Delay**; [618] **Remote Jitter**; [619] **Remote SSRC Sender**; [620] **Local RTP IP**; [621] **Local RTP Port**; [622] **Remote RTP IP**; [623] **Remote RTP Port**; [624] **RTP IP DiffServ**; [625] **Local R Factor**; [626] **Remote R Factor**; [627] **Local MOS CQ**; [628] **Remote MOS CQ**; [629] **AMD Decision**; [630] **AMD Decision Probability**; [631] **Latched RTP IP**; [632] **Latched RTP Port**; [633] **Latched T38 IP**; [634] **Latched T38 Port**; [635] **Coder Transcoding** **Note:** For 'CDR Types' **Syslog SBC**, **Local Storage SBC**, and **RADIUS SBC**, the above media-related CDR fields are added only to "CALL_END" SBC signaling CDR report Types (which by default, include only signaling CDR fields). ■ **Syslog SBC** (signaling), **Local Storage SBC**, **RADIUS SBC**, and **JSON SBC**: [800] **Source URI**; [801] **Destination URI**; [802] **Source URI Before Manipulation**; [803] **Destination URI Before Manipulation**; [804] **Redirect URI**; [805] **Redirect URI Before Manipulation**; [806] **SIP Method**; [807] **Direct Media**; [808] **Source Username**; [809] **Destination Username**; [810] **Source Username Before Manipulation**; [811] **Destination Username Before Manipulation**; [812] **Source Host**; [813] **Destination Host**; [814] **Source Host Before Manipulation**; [815] **Destination Host Before Manipulation**; [816] **Source Dial Plan Tags**; [817] **Destination Dial Plan Tags**; [818] **Remote SIP User Agent**; [819] **Media List**; [820] **Voice AI Connector ID**; [821] **Voice AI Connector Name**; [822] **Is Recorded**
'Title' `title` [SBCCDRFormat_Title]	Defines a new name for the CDR field (for Syslog or local storage) or for the RADIUS Attribute prefix name (for RADIUS accounting) that you selected in the 'Column Type' parameter. The valid value is a string of up to 31 characters. You can also configure the name to be enclosed by quotation marks (single or double). For example, if you want the CDR field name to appear as 'Phone Duration', you must configure the parameter to 'Phone Duration'. You can also configure the CDR field name with an equals (=) sign, for example "call-connect-time=". **Note:**

Parameter	Description
	■ For VSA's that do not require a prefix name, leave the parameter undefined. ■ The parameter's value is case-sensitive. For example, if you want the CDR field name to be Phone-Duration, you must configure the parameter to "Phone-Duration" (i.e., uppercase "P" and "D").
'RADIUS Attribute Type' `radius-type` [SBCCDRFormat_RadiusType]	Defines whether the RADIUS Attribute of the CDR field is a standard or vendor-specific attribute. ■ [0] **Standard** = (Default) For standard RADIUS Attributes. ■ [1] **Vendor Specific** = For vendor-specific RADIUS Attributes (VSA). **Note:** The parameter is applicable only to RADIUS accounting (i.e., 'CDR Type' parameter configured to **RADIUS SBC**).
'RADIUS Attribute ID' `radius-id` [SBCCDRFormat_RadiusID]	Defines an ID for the RADIUS Attribute. For VSAs, this represents the VSA ID; for standard Attributes, this represents the Attribute ID (first byte of the Attribute). The valid value is 0 to 255 (one byte). The default is 0. **Note:** ■ The parameter is applicable only to RADIUS accounting (i.e., 'CDR Type' parameter configured to **RADIUS SBC**). ■ For VSA's (i.e., 'RADIUS Attribute Type' parameter configured to **Vendor Specific**), the parameter must be configured to any value other than 0. ■ For standard RADIUS Attributes (i.e., 'RADIUS Attribute Type' parameter configured to **Standard**), the value **must** be a "known" RADIUS ID (per RFC for RADIUS). However, if you configure the ID to 0 (default) for any of the RADIUS Attributes (configured in the 'Column Type' parameter) listed below and then apply your rule (Click **Apply**), the device automatically replaces the value with the RADIUS Attribute's ID according to the RFC: ✓ **Destination Username:** 30 ✓ **Source Username:** 31 ✓ **Accounting Status Type:** 40 ✓ **Local Input Octets:** 42 ✓ **Local Output Octets:** 43

Parameter	Description
	✓ **Call Duration:** 46 ✓ **Local Input Packets:** 47 ✓ **Local Output Packets:** 48 If you configure the value to 0 and the RADIUS Attribute is not any of the ones listed above, the configuration is invalid.

Example of Call Variables for CDR Customization

This section provides an example of using a call variable to customize the Syslog SBC (signaling) CDR. The example includes the following:

- Uses the call variable `Var.Call.Src.UserDefined1` in a Message Manipulation rule to store the value of the SIP header "X-AccountNumber" received in a 200 OK response.
- Customizes the SBC CDR format to add a CDR field titled "Account" whose value is obtained from the call variable, `Var.Call.Src.UserDefined1` used in the Message Manipulation rule.

➢ **To customize CDR using a call variable:**

1. In the Message Manipulations table (see Configuring SIP Message Manipulation on page 634), configure the following rule:
   - 'Index': **0**
   - 'Name': **Store X-AccountNumber header**
   - 'Manipulation Set ID': **0**
   - 'Message Type': **Invite.Response.2xx**
   - 'Action Subject': **Var.Call.Src.UserDefined1**
   - 'Action Type': **Modify**
   - ' Action Value': **Header.X-AccountNumber**
2. In the SBC CDR Format table, configure the following rule:
   - 'Index': **0**
   - 'CDR Type': **Syslog SBC**
   - 'Field Type': **Var Call User Defined 1**
   - 'Title': **Account**

The following shows an example of a received SIP 200 OK response message with the X-InContact-BusNo header:

```
SIP/2.0 200 OK
Via: SIP/2.0/UDP 172.28.244.31:5060;branch=z9hG4bKac166782921
Contact: <sip:Stack@10.21.19.151:5060>
To: <sip:+18017155444@abc.com>;tag=87245f3d
From: "usera"<sip:usera@abc.com>;tag=1c1187059515
Call-ID: 628022773122202022133@172.28.244.31
CSeq: 1 INVITE
Session-Expires: 1200;refresher=uas
Content-Type: application/sdp
Supported: timer
X-AccountNumber: 87654321
```

The following shows the generated CDR:

```
|Orig |Account|
|LCL |87654321 |
```

Customizing CDR Indication for Call Success or Failure based on Responses

The CDR can indicate if a call was successful ("yes") or a failure ("no"), using the 'Call Success' CDR field. This is an optional field that you can include in CDRs, by customizing the CDR format (as described in Customizing CDRs for SBC Calls).

The device determines if a call is a success or failure based on the release (termination) reason of the call, which can be a SIP response code received from the SIP User Agent or an internal response generated by the device. However, you can change the device's default mapping of call success and failure with these responses. For example, by default, the device considers a call that is released with a SIP 486 (Busy Here) response as a call failure (i.e., 'Call Success' CDR field displays "no"). Using this feature, you can configure the device to consider SIP 486 (Busy Here) responses as call success.

➢ **To customize CDR indication for call success and failure:**

1. Open the Call Detail Record Settings page (**Troubleshoot** menu > **Troubleshoot** tab > **Call Detail Record** folder > **Call Detail Record Settings**).
2. To customize call success or failure indication for SIP reasons:
   - In the 'Call Success SIP Reasons' [CallSuccessSIPReasons] field, configure the SIP response codes that you want the device to consider as call success.
   - In the 'Call Failure SIP Reasons' [CallFailureSIPReasons] field, configure the SIP response codes that you want the device to consider as call failure.

Call Success SIP Reasons

Call Failure SIP Reasons

If your configuration results in overlapping reasons between the two parameters above, preference is given to the parameter with the specific response code instead of the parameter with the range ("xx"). For example, if 'Call Success SIP Reasons' is configured to "486,5xx" and 'Call Failure SIP Reasons' to "502", a call with SIP response code 502 is considered a call failure, as the 'Call Failure SIP Reasons' parameter is configured with the specific code while the 'Call Success SIP Reasons' is configured with the code range (5xx).

3. To customize call success or failure indication for internal reasons:
   - In the 'Call Success Internal Reasons' [CallSuccessInternalReasons] field, configure the internal response codes that you want the device to consider as call success.
   - In the 'Call Failure Internal Reasons' [CallFailureInternalReasons] field, configure the internal response codes that you want the device to consider as call failure.

Call Success Internal Reasons

Call Failure Internal Reasons

For a list of the internal response codes, see the 'Termination Reason' [410] CDR field in CDR Field Description on page 990.

4. To customize call success or failure indication for the internal response "GWAPP_NO_USER_RESPONDING" (18) before or after call connect (SIP 200 OK):
   - From the 'No User Response Before Connect' [NoUserResponseBeforeConnectSuccess] drop-down list, select **Call Failure** if you want the device to consider a call as a failure when this response is received before call connect.
   - From the 'No User Response After Connect' [NoUserResponseAfterConnectSuccess] drop-down list, select **Call Success** if you want the device to consider a call as a success when this response is received after call connect.

No User Response Before Connect Call Success

No User Response After Connect Call Failure

5. To customize call success or failure indication for the internal response 'RELEASE_BECAUSE_CALL_TRANSFERRED' (807) before or after call connect (SIP 200 OK):

- From the 'Call Transferred before Connect' [CallTransferredBeforeConnectSuccess] drop-down list, select **Call Success** if you want the device to consider the call as a success when this response is received before call connect (SIP 200 OK).
- From the 'Call Transferred after Connect' [CallTransferredAfterConnectSuccess] drop-down list, select **Call Failure** if you want the device to consider the call as a failure when this response is received after call connect (SIP 200 OK).

Hiding Caller and Callee CDR Field Values

You can enable the device to hide (using an * asterisk) the values of the Caller and Callee fields in CDRs that are displayed by the following:

- Web interface:
  - SBC CDR History table (see Viewing CDR History of SBC and Test Calls on page 970)
- CLI (refer to the *CLI Reference Guide*):
  - `show voip calls history`
  - `show voip calls active`

➢ **To hide Caller and Callee CDR field values:**

- ini file:

```
CDRHistoryPrivacy = 1
```

- CLI:

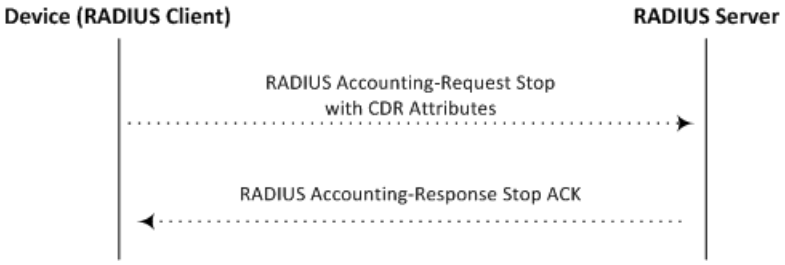
```
(config-troubleshoot)# cdr
(cdr)# cdr-history-privacy hide-caller-and-callee
```

Configuring RADIUS Accounting

The device supports RADIUS Accounting (per RFC 2866) and sends accounting data of SIP calls as call detail records (CDR) to a RADIUS Accounting server. CDR-based accounting messages can be sent upon call release, call connection and release, or call setup and release. This section lists the CDR attributes for RADIUS accounting.

The following figure shows the interface between the device and the RADIUS server, based on the RADIUS Accounting protocol. For each CDR that the device sends to the RADIUS server, it sends an Accounting-Request Stop with all the CDR attributes. When the RADIUS server

successfully receives all the CDR attributes, it responds with an Accounting-Response Stop ACK message to the device. If the device does not receive the Accounting-Response ACK message, it can resend the Accounting-Request Stop with all CDR attributes again, up to a user-defined number of re-tries (see Configuring RADIUS Packet Retransmission).

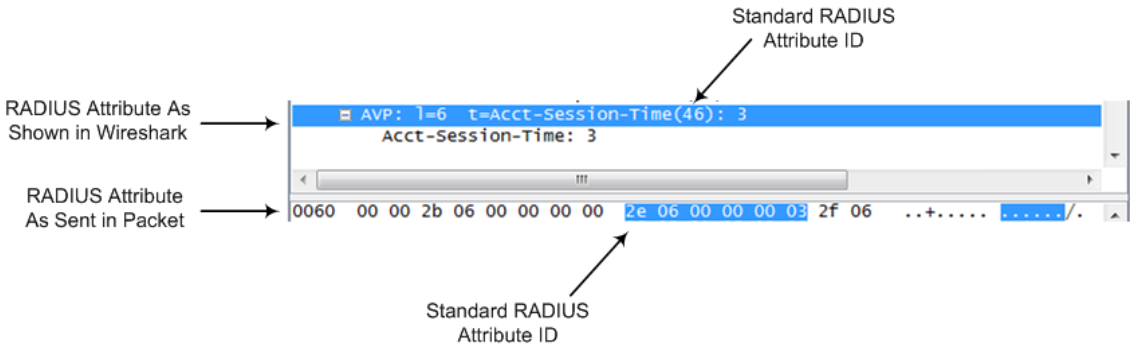


There are two types of data that can be sent to the RADIUS server. The first type is the accounting-related attributes and the second type is the vendor specific attributes (VSA):

- **Standard RADIUS Attributes (per RFC):** A typical standard RADIUS attribute is shown below. The RADIUS attribute ID depends on the attribute.

```
2e 06 00 00 00 03 --- Data
|  |
|  Length (including header)
RADIUS ID
```

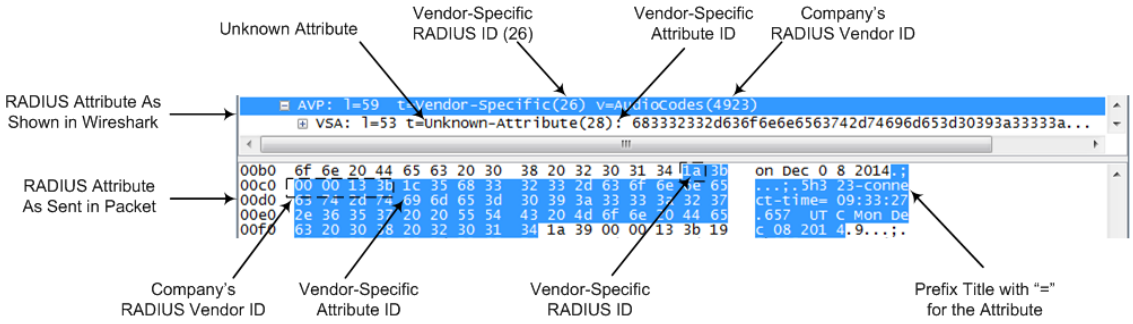
  The following figure shows a standard RADIUS attribute collected by Wireshark. The bottom pane shows the RADIUS attribute information as sent in the packet; the upper pane is Wireshark's interpretation of the RADIUS information in a more readable format. The example shows the attribute in numeric format (32-bit number in 4 bytes).



- **Vendor-specific RADIUS Attributes:** RADIUS attributes that are specific to the device (company) are referred to as Vendor-specific attributes (VSA). The CDR of VSAs are sent with a general RADIUS ID of 26 to indicate that they are vendor-specific (non-standard). In addition, the company's registered vendor ID (as registered with the Internet Assigned Numbers Authority or IANA) is also included in the packet. The device's default vendor ID is 5003, which can be changed (see Configuring the RADIUS Vendor ID). The VSA ID is also included in the packet.

```
1a 13 00 00 13 8b 21 0d 68 33 32 33 2d 67 77 2d 69 64 3d --- Data
|  |  |  |  |  |  |  |
|  |  Vendor ID    |  Vendor part length
|  |  (5003)       Vendor-Specific Attribute (VSA) ID
|  Length (including header)
RADIUS ID indicating vendor-specific (26)
```

The following figure shows a vendor-specific RADIUS attribute collected by Wireshark. The bottom pane shows the RADIUS attribute information as sent in the packet; the upper pane is Wireshark's interpretation of the RADIUS information in a more readable format. The example shows the attribute in string-of-characters format.



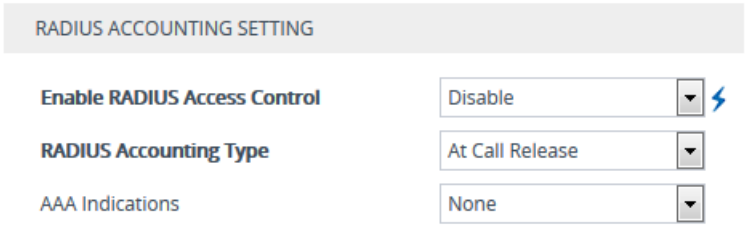
You can customize the prefix title of the RADIUS attribute name and ID. For more information, see Customizing CDRs for SBC Calls .

To configure the address of the RADIUS Accounting server, see Configuring RADIUS Servers. For all RADIUS-related configuration, see RADIUS-based Services.

➢ To configure RADIUS accounting:

1. Open the Call Detail Record Settings page (**Troubleshoot** menu > **Troubleshoot** tab > **Call Detail Record** folder > **Call Detail Record Settings**).
2. Configure the following parameters:
   - From the 'Enable RADIUS Access Control' [EnableRADIUS] drop-down list, select **Enable**.
   - From the 'RADIUS Accounting Type' [RADIUSAccountingType] drop-down list, select the stage of the call that RADIUS accounting messages are sent to the RADIUS accounting server.
   - From the 'AAA Indications' [AAAIndications] drop-down list, select whether you want Authentication, Authorization and Accounting (AAA) indications.

   For a detailed description of the parameters, see RADIUS Parameters.



3. Click **Apply**, and then reset the device with a save-to-flash for your settings to take effect.

The table below lists the RADIUS Accounting CDR attributes included in the communication packets transmitted between the device and a RADIUS server.

Table 50-4: Default RADIUS Accounting CDR Attributes

Attribute ID	Attribute Name	Vendor-Specific Attribute (VSA) ID	Description	Value Format	Example	AAA
Request Attributes						
1	user-name	(Standard)	Account number or calling party number or blank	String up to 15 digits long	5421385747	Start Acc Stop Acc
4	nas-ip-address	(Standard)	IP address of the requesting device	Numeric	192.168.14.43	Start Acc Stop Acc
6	service-type	(Standard)	Type of service requested	Numeric	1: login	Start Acc Stop Acc
26	h323-incoming-conf-id	1	SIP call identifier	Up to 32 octets	h323-incoming-conf-id=38393530	Start Acc Stop Acc
26	h323-remote-address	23	IP address of the remote gateway	Numeric	-	Stop Acc

Attribute ID	Attribute Name	Vendor-Specific Attribute (VSA) ID	Description	Value Format	Example	AAA
26	h323-conf-id	24	H.323/SIP call identifier	Up to 32 octets	-	Start Acc Stop Acc
26	h323-setup-time	25	Setup time in NTP format 1	String	h323-setup-time=09:33:26.621 Mon Dec 2014	Start Acc Stop Acc
26	h323-call-origin	26	Originator of call: ■ "answer": Call originated from the incoming leg ■ "originate": Call originated from the outgoing leg	String	h323-call-origin=answer	Start Acc Stop Acc
26	h323-call-type	27	Protocol type or family used on this leg of the call. The value is always "VOIP".	String	h323-call-type=VOIP	Start Acc Stop Acc
26	h323-connect-time	28	Connect time in NTP format	String	h323-connect-time=09:33:37.657 UTC Mon Dec 08 2015	Stop Acc
26	h323-disconnect-time	29	Disconnect time in NTP format	String	-	Stop Acc

Attribute ID	Attribute Name	Vendor-Specific Attribute (VSA) ID	Description	Value Format	Example	AAA
26	h323-disconnect-cause	30	Disconnect cause code (Q.850)	Numeric	h323-disconnect-cause=16	Stop Acc
26	h323-gw-id	33	Name of the gateway	String	h323-gw-id=<SIP ID string>	Start Acc Stop Acc
26	sip-call-id	34	SIP Call ID	String	sip-call-id=abcde@ac.com	Start Acc Stop Acc
26	call-terminator	35	Terminator of the call: ■ "yes": Call terminated by the outgoing leg ■ "no": Call terminated by the incoming leg	String	call-terminator=yes	Stop Acc
26	terminator	37	Terminator of the call: ■ "answer": Call originated from the incoming leg ■ "originate": Call originated from the	String	terminator=originate	Stop Acc

Attribute ID	Attribute Name	Vendor-Specific Attribute (VSA) ID	Description	Value Format	Example	AAA
			outgoing leg			
30	called-station-id	(Standard)	Destination URI	String	8004567145	Start Acc
31	calling-station-id	(Standard)	Source URI	String	5135672127	Start Acc Stop Acc
40	acct-status-type	(Standard)	Account Request Type: ■ "1" (start): Sent in Call Start or Call Connect CDRs ■ "2" (stop): Sent in Call End CDRs only. **Note:** It is highly recommended to add this attribute if you are customizing the RADIUS CDR format (see Customizing CDRs for SBC Calls and Test Calls on page 1036 for SBC calls).	Numeric	1	Start Acc Stop Acc
41	acct-delay-time	(Standard)	No. of seconds tried in sending a particular record	Numeric	5	Start Acc

Attribute ID	Attribute Name	Vendor-Specific Attribute (VSA) ID	Description	Value Format	Example	AAA
						Stop Acc
42	acct-input-octets	(Standard)	Number of octets received for that call duration (applicable only if media anchoring)	Numeric	-	Stop Acc
43	acct-output-octets	(Standard)	Number of octets sent for that call duration (applicable only if media anchoring)	Numeric	-	Stop Acc
44	acct-session-id	(Standard)	A unique accounting identifier that corresponds to the acct-status-type attribute and thus, the identifier of the start CDR is identical to the stop CDR ID of the same call. This attribute is composed of the Session ID of the call (e.g., [SID=9be7fc:152:99757]) followed by a colon (:) and the leg ID (e.g., 9be7fc:152:99757:1 for the SBC incoming leg or Gateway call,	String	34832	Start Acc Stop Acc

Attribute ID	Attribute Name	Vendor-Specific Attribute (VSA) ID	Description	Value Format	Example	AAA
			and 9be7fc:152:99757:2 for the SBC outgoing leg).			
46	acct-session-time	(Standard)	For how many seconds the user received the service	Numeric	-	Stop Acc
47	acct-input-packets	(Standard)	Number of packets received during the call	Numeric	-	Stop Acc
48	acct-oputput-packets	(Standard)	Number of packets sent during the call	Numeric	-	Stop Acc
61	nas-port-type	(Standard)	Physical port type of device on which the call is active	String	0: Asynchronous	Start Acc Stop Acc
Response Attributes						
26	h323-return-code	103	The reason for failing authentication (0 = ok, other number failed)	Numeric	0 Request accepted	Stop Acc
44	acct-session-id	(Standard)	A unique accounting identifier – match start & stop	String	-	Stop Acc

Below is an example of RADIUS Accounting, where non-standard parameters are preceded with brackets:

```
Accounting-Request (4)
user-name = 111
acct-session-id = 1
nas-ip-address = 212.179.22.213
nas-port-type = 0
acct-status-type = 2

acct-session-time = 1
acct-input-packets = 122
acct-output-packets = 220
called-station-id = 201
calling-station-id = 202
```

```
// Accounting non-standard parameters:
(4923 33) h323-gw-id =
(4923 23) h323-remote-address = 212.179.22.214
(4923 1) h323-ivr-out = h323-incoming-conf-id:02102944 600a1899 3fd61009
0e2f3cc5
(4923 30) h323-disconnect-cause = 22 (0x16)
(4923 27) h323-call-type = VOIP
(4923 26) h323-call-origin = Originate
(4923 24) h323-conf-id = 02102944 600a1899 3fd61009 0e2f3cc5
```

51 Remote Monitoring of Device behind NAT

When the device is located behind a NAT, you can configure it to periodically send monitoring reports to a third-party, remote HTTP-based monitoring server. This third-party server is configured on the device as a Remote Web Service (HTTP host), where the 'Type' parameter is set to **Remote Monitoring**. The device sends the reports over HTTP/S using RESTful API (in JSON format), where the device acts as the client.

You can choose to send various reports to the monitoring server:

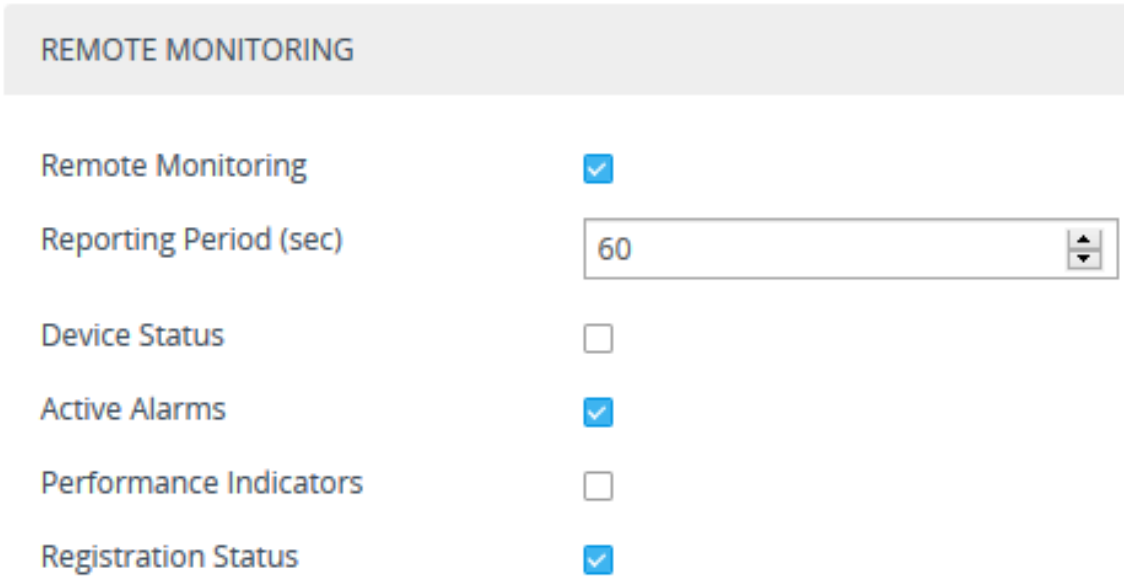
- **Status reports:** These reports contain status information of the device, for example, software version, network configuration (IP network interfaces, Ethernet port interfaces, and proxy addresses), IP Groups, and serial number).
- **Active alarms reports:** These reports contain currently active alarms.
- **Key performance indicators reports:** These reports contain performance monitoring statistics, for example, number of active SBC sessions, average call duration, and number of established inbound calls.
- **Registration status reports:** These reports contain status information of SIP User Agents (UA) currently registered with the device.

If the device receives an HTTP failure response (4xx/5xx/6xx) from the Remote Web Service when it attempts to send it a monitoring report, the device raises the SNMP alarm, acRemoteMonitoringAlarm (with Warning severity level). This alarm is cleared only when it receives an HTTP successful response (2xx) from the server.

- Currently, you can configure the device to send monitoring reports to only one Remote Web Service.
- If the report contains the **more** attribute with value "True", it means that the report has reached its maximum file size and the device will send another report with more information. The last report doesn't contain this attribute.

➢ **To enable remote monitoring of device behind NAT:**

1. In the Remote Web Services table (see Configuring Remote Web Services on page 316), configure a Remote Web Service with the 'Type' parameter set to **Remote Monitoring**.
2. Open the Web Service Settings page (**Setup** menu > **IP Network** tab > **Web Services** folder > **Web Service Settings**).



3. Select the 'Remote Monitoring' check box to enable the feature.
4. In the 'Reporting Period' field, configure the interval (in seconds) between each sent report.
5. Select the check boxes of the corresponding report types (information) that you want the device to send:
   - 'Device Status': Report contains status information of the device
   - 'Active Alarms': Report contains currently active alarms
   - 'Performance Indicators': Report contains performance monitoring statistics
   - 'Registration Status': Report contains information of users registered with the device
6. Click **Apply**.

Part IX

Diagnostics

52 Syslog and Debug Recording

For debugging and troubleshooting, you can use the device's Syslog and/or Debug Recording capabilities:

- **Syslog:** Syslog is an event notification protocol that enables a device to send event notification messages across IP networks to event message collectors, also known as Syslog servers. The device contains an embedded Syslog client, which sends error reports / events that it generates to a remote Syslog server using the IP / UDP protocol. This information is a collection of error, warning, and system messages that records every internal operation of the device.
- **Debug Recording:** The device can send debug recording packets to a debug capturing server. When the debug recording is activated, the device duplicates all messages that are sent and/or received by it and then sends them to an external server defined by IP address. The debug recording can be done for different types of traffic such as RTP/RTCP, T.38, and SIP. Debug recording is used for advanced debugging when you need to analyze internal messages and signals. Debug recording is also useful for recording network traffic in environments where hub or port mirroring is unavailable and for recording internal traffic between two endpoints on the same device.

You can include Syslog messages in debug recording (see Configuring Log Filter Rules).

Configuring Log Filter Rules

The Logging Filters table lets you configure up to 60 rules for filtering debug recording packets, Syslog messages, and Call Detail Records (CDR). The log filter determines the calls for which you want to generate debug recording packets, Syslog messages or CDRs. For example, you can add a rule to generate Syslog messages only for calls belonging to IP Groups 2 and 4, or for calls belonging to all IP Groups except IP Group 3.

You can also configure log filters for generating CDRs only and saving them on the device (local storage). Debug recording log filters can include signaling information (such as SIP messages), Syslog messages, CDRs, media (RTP, RTCP, and T.38), and pulse-code modulation (PCM) .

If you don't configure any rules in the Logging Filters table and you have globally enabled debug recording (by configuring the Debug Recording server's address - see Note below), Syslog (global parameter - see Note below), and/or CDR generation (global parameter for enabling Syslog - see Note below), logs are generated for all calls. Thus, the benefit of log filtering is that it allows you to create logs per specific calls, eliminating the need for additional device resources (CPU consumption) otherwise required when logs are generated for all calls.

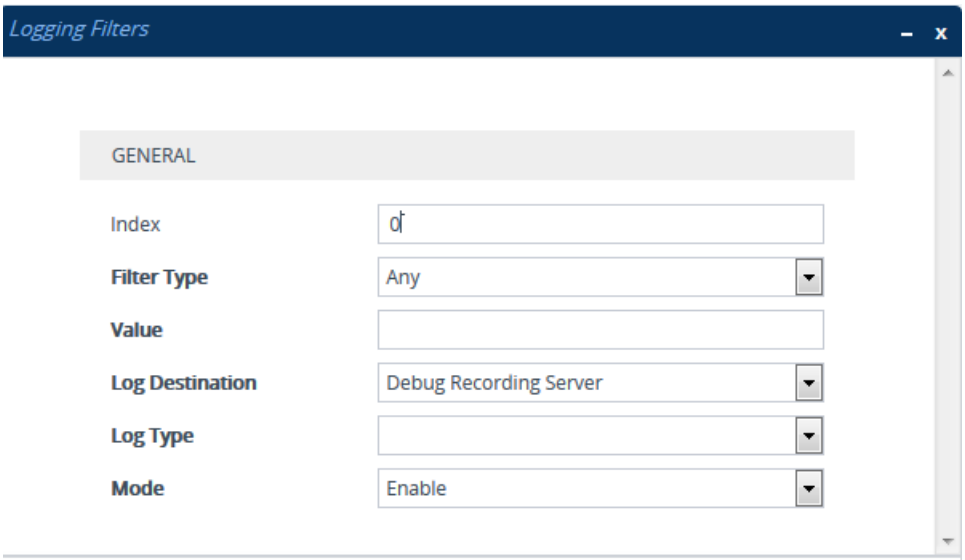
You can enable and disable configured Log Filter rules. Enabling a rule activates the rule, whereby the device starts generating the debug recording packets, Syslog messages, or CDRs. Disabling a rule is useful, for example, if you no longer require the rule, but may need it in the future. Thus, instead of deleting the rule entirely, you can simply disable it.

- If you want to configure a Log Filter rule that logs Syslog messages to a Syslog server (i.e., not to a Debug Recording server), you must enable Syslog functionality, using the 'Enable Syslog' (EnableSyslog) parameter (see Enabling Syslog). Enabling Syslog functionality is not required for rules that include Syslog messages in the debug recording sent to the Debug Recording server.
- To configure the Syslog server's address, see Configuring the Syslog Server Address. To configure additional, global Syslog settings, see Configuring Syslog.
- To configure the Debug Recording server's address, see Configuring the Debug Recording Server Address.
- To configure additional, global CDR settings such as at what stage of the call the CDR is generated (e.g., start and end of call), see Configuring CDR Reporting.

The following procedure describes how to configure Log Filter rules through the Web interface. You can also configure it through ini file [LoggingFilters] or CLI (`configure troubleshoot > logging logging-filters`).

➢ To configure a Log Filter rule:

1. Open the Logging Filters table (**Troubleshoot** menu > **Troubleshoot** tab > **Logging** folder > **Logging Filters**).
2. Click **New**; the following dialog box appears:



3. Configure a Log Filtering rule according to the parameters described in the table below.
4. Click **Apply**.

Table 52-1: Logging Filters Table Parameter Descriptions

Parameter	Description
'Index' [LoggingFilters_Index]	Defines an index number for the new table row. **Note:** Each row must be configured with a unique index.

Parameter	Description
'Filter Type' `filter-type` [LoggingFilters_ FilterType]	Defines the filter type criteria. ■ [1] **Any**= (Default) Debug recording is done for all calls. ■ [8] **IP Group** = Filters the log by IP Group. To configure IP Groups, see Configuring IP Groups. ■ [9] **SRD** = Filters the log by SRD. To configure SRDs, see Configuring SRDs. ■ [10] **Classification** = Filters the log by Classification rule. To configure Classification rules, see Configuring Classification Rules. ■ [11] **IP-to-IP Routing** = Filters the log by IP-to-IP Routing rule. To configure IP-to-IP Routing rules, see Configuring SBC IP-to-IP Routing Rules. ■ [12] **User** = Filters the log by user. The user is defined by username or username@hostname in the Request-URI of the SIP Request-Line. For example, "2222@10.33.45.201" (without quotation marks), which represents the following INVITE: `INVITE sip:2222@10.33.45.201;user=phone SIP/2.0` ■ [13] **IP Trace** = Filters the log by an IP network trace, using Wireshark-like expressions. For more information, see Filtering IP Network Traces. ■ [14] **SIP Interface** = Filters the log by SIP Interface. To configure SIP Interfaces, see Configuring SIP Interfaces.
'Value' `value` [LoggingFilters_Value]	Defines the value for the filtering type configured in the 'Filter Type' parameter. The value can include the following: ■ A single value. ■ A range, using a hyphen "-" between the two values. For example, to specify IP Groups 1, 2 and 3, configure the parameter to "1-3" (without quotation marks). ■ Multiple, non-contiguous values, using commas "," between each value. For example, to specify IP Groups 1, 3 and 9, configure the parameter to "1,3,9" (without quotation marks). ■ To exclude specific configuration entities from the log filter,

Parameter	Description
	use the exclamation (**!**) wildcard character. For example, to include all IP Groups in the filter except IP Group ID 2, configure the 'Filter Type' parameter to **IP Group** and the 'Value' parameter to "!2" (without quotation marks). **Note:** For SBC calls, a Logging Filter rule applies to the entire session (i.e., inbound and outbound legs). Therefore, if you want to exclude logging of specific calls, you need to configure the 'Value' parameter with both legs. For example: ✓ If you want to exclude logs for calls between IP Group 1 and IP Group 2, configure the parameter to "!1,2" (without quotation marks). ✓ If you want to exclude logs for calls between SIP Interface 4 and SIP Interface 9, configure the parameter to "!4,9" (without quotation marks). **Note:** ■ You can use the index number or string name to specify the configuration entity for the following 'Filter Types': **IP Group**, **SRD**, **Classification**, **IP-to-IP Routing**, or **SIP Interface**. For example, to specify IP Group at Index 2 with the name "SIP Trunk", configure the parameter to either "2" or "SIP Trunk" (without quotation marks). ■ For IP trace expressions, see Filtering IP Network Traces.
'Log Destination' `log-dest` [LoggingFilters_ LogDestination]	Defines where the device sends the log file. ■ [0] **Syslog Server** = The device generates Syslog messages based on the configured log filter and sends them to a user-defined Syslog server. ■ [1] **Debug Recording Server** = (Default) The device generates debug recording packets based on the configured log filter and sends them to a user-defined Debug Recording server. ■ [2] **Local Storage** = The device generates CDRs based on the configured log filter and stores them locally on the device. For more information on local CDR storage, see Storing CDRs on the Device ■ [3] **Call Flow Server** = The device sends SIP messages to a call flow server (i.e., OVOC) for displaying SIP call dialog sessions as SIP call flow diagrams. For this functionality, you also need to configure the 'Log Type' parameter to **Call Flow**. For

Parameter	Description
	enabling this functionality, see Enabling SIP Call Flow Diagrams in OVOC on page 1097. **Note:** ■ If you configure the parameter to **Syslog Server**: ✓ If you have also configured the debug level to **No Debug** (see the [GwDebugLevel] parameter in Configuring Syslog Debug Level), the syslog messages include only system warnings and errors. ✓ The 'Log Type' parameter (below) is not applicable (all syslog messages are sent to the syslog server). ■ If the 'Filter Type' parameter is configured to **IP Trace**, you must configure the parameter to **Debug Recording Server**. ■ If you configure the parameter to **Local Storage**, you must configure the 'Log Type' parameter to **CDR Only**. ■ If you configure the parameter to **Syslog Server** and the debug level, using the [GwDebugLevel] parameter, is configured to **No Debug** (see Configuring Syslog Debug Level), the Syslog messages include only system Warnings and Errors. ■ If you configure the parameter to **Debug Recording Server**, you can also include Syslog messages in the debug recording packets sent to the debug recording server. To include Syslog messages, configure the 'Log Type' parameter (see below) to the relevant option.
'Log Type' `log-type` [LoggingFilters_ CaptureType]	Defines the type of messages to include in the log file. ■ [0] = (Default) Not configured. The option is applicable only for sending Syslog messages to a Syslog server (i.e., 'Log Destination' parameter is configured to **Syslog Server**). ■ [1] **Signaling** = The option is applicable only to debug recording (i.e., 'Log Destination' parameter is configured to **Debug Recording Server**). The debug recording includes signaling information such as SIP signaling messages, Syslog messages, CDRs, and the device's internal processing messages. ■ [2] **Signaling & Media** = The option is applicable only to debug recording (i.e., 'Log Destination' parameter is configured to **Debug Recording Server**). The debug recording

Parameter	Description
	includes media (RTP/RTCP/T.38), and only signaling and Syslog messages associated with the recorded media. **Note:** The device requires a lot of resources for media debug recording. The number of media sessions (and associated signaling) that the device records depends on available resources. Therefore, when many media sessions need to be recorded (e.g., when the 'Filter Type' parameter is configured to **Any**) not all media sessions (and associated signaling) are recorded. If the device has no resources to debug record any media, it doesn't debug record any signaling as well. As debug recording of signaling requires less resources than media debug recording, if you want to perform debug recording only on signaling, then it is recommended to configure the parameter to **Signaling**. ■ [3] **Signaling & Media & PCM** = The option is applicable only to debug recording (i.e., 'Log Destination' parameter is configured to **Debug Recording Server**). The debug recording includes signaling, Syslog messages, media, and PCM. ■ [5] **CDR Only** = Only CDRs are generated. The option is applicable only if the 'Log Destination' parameter is configured to **Syslog Server** or **Local Storage**. When configured to **Syslog Server**, only CDRs are included in the Syslog messages (excluding all system logs and alerts) sent to the Syslog server. ■ [6] **Call Flow** = The device sends SIP messages (in XML format), as they occur in real-time, to OVOC for displaying SIP call dialog sessions as call flow diagrams. For this functionality, you also need to configure the 'Log Destination' parameter to **Call Flow Server**. For enabling this functionality, see Enabling SIP Call Flow Diagrams in OVOC on page 1097. ■ [7] **SIP Only** = The option is applicable only to debug recording (i.e. the 'Log Destination' parameter is configured to **Debug Recording Server** or **Syslog Server**). The debug recording includes only SIP messages. **Note:** ■ If you configure the 'Log Destination' parameter to **Local Storage**, you must configure the 'Log Type' parameter to **CDR Only**.

Parameter	Description
	■ The parameter is not applicable when the 'Filter Type' parameter is configured to **IP Trace**. ■ To include Syslog messages in debug recording, it is unnecessary to enable Syslog functionality.
'Mode' mode [LoggingFilters_Mode]	Enables and disables the rule. ■ [0] **Disable** ■ [1] **Enable** (default)

Filtering IP Network Traces

You can filter Syslog and debug recording messages for IP network traces, by configuring the 'Filter Type' parameter to **IP Trace** in the Logging Filters table. IP traces record any IP stream, according to destination and/or source IP address, or port and Layer-4 protocol (UDP, TCP or any other IP type as defined by http://www.iana.com). Network traces are typically used to record HTTP.

When the **IP Trace** option is selected, only the 'Value' parameter is applicable ('Syslog' and 'Capture Type' parameters are not applicable). The 'Value' parameter configures the Wireshark-like filtering expressions for your IP trace. The following Wireshark- like expressions are supported:

Table 52-2: Supported Wireshark-like Expressions for 'Value' Parameter

Expression	Description
ip.src	Defines the source IPv4 address.
ipv6.src	Defines the source IPv6 address.
ip.dst	Defines the destination IPv4 address.
ipv6.dst	Defines the destination IPv6 address.
ip.addr	Defines IPv4 addresses (up to two).
ipv6.addr	Defines IPv6 addresses (up to two).
ip.proto	Defines the IP protocol type (PDU), entered as an enumeration value (e.g., 1 is ICMP, 6 is TCP, 17 is UDP).
udp, tcp, icmp, sip, ldap, http, https	Defines single expressions for the protocol type.
udp.port, tcp.port	Defines the transport layer.

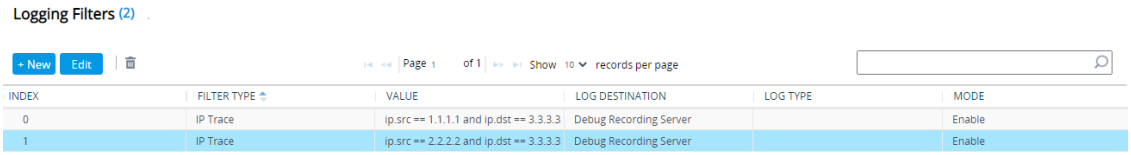
Expression	Description
udp.srcport, tcp.srcport	Defines the transport layer for the source port.
udp.dstport, tcp.dstport	Defines the transport layer for the destination port.
and, &&, ==, <, >	Comparison operators used between expressions.

Below are examples of configured expressions for the 'Value' parameter:

- udp && ip.addr==10.8.6.55
- ip.src==10.8.6.55 && udp.port>=5000 and udp.port<6000
- ip.dst==10.8.0.1/16
- ip.addr==10.8.6.40
- ipv6.addr==2001:0db8:85a3:0000:0000:8a2e:0370:7334
- ipv6.src==2001:db8:abcd:0012::0/64

For conditions requiring the "or" / "||" expression, add multiple rows in the Logging Filters table. For example, the Wireshark condition "(ip.src == 1.1.1.1 or ip.src == 2.2.2.2)" and "ip.dst == 3.3.3.3" can be done by adding two rows in the table, where the 'Value' parameter of each row has the following value:

- Index #0: 'Value' parameter is configured to (without quotation marks) "ip.src == 1.1.1.1 and ip.dst == 3.3.3.3"
- Index #1: 'Value' parameter is configured to (without quotation marks) "ip.src == 2.2.2.2 and ip.dst == 3.3.3.3"



- If you leave the 'Value' parameter empty, the device records all IP traffic types.
- You cannot configure the 'Value' parameter with IPv4 addresses together with IPv6 addresses.
- You cannot configure the 'Value' parameter with ip.addr or udp/tcp.port together with ip.src/dst or udp/tcp.srcport/dstport. For example, "ip.addr==1.1.1.1 and ip.src==2.2.2.2" (without quotation marks) is an invalid configuration value.
- You cannot configure the 'Value' parameter with ipv6.addr or udp/tcp.port together with ipv6.src/dst or udp/tcp.srcport/dstport. For example, "ipv6.addr==2001:0db8:85a3:0000:0000:8a2e:0370:7334 and ipv6.src==2001:db8:abcd:0012::0/64" (without quotation marks) is an invalid configuration value.

Configuring Syslog

This section describes how to configure Syslog. To filter Syslog messages, see Configuring Log Filter Rules.

Syslog Message Format

The Syslog message is sent from the device to a Syslog server as an ASCII (American Standard Code for Information Interchange) message. Syslog uses UDP as its underlying transport layer mechanism. By default, UDP port 514 is assigned to Syslog, but this can be changed (see Enabling Syslog).

Syslog includes two types of log messages:

- **SIP Call Session Logs:** Logs relating to call sessions (e.g., call established). These logs are identified by a session ID ("SID"), described in detail in the table below. For example:

```
10:44:11.299 10.15.77.55 local0.notice [S=511941] [SID=50dcb2:31:12079]
(N 483455) ReleaseAddress. IPv4IF=1 IPv6IF=-1 Port=7500 [Time:10-
09@09:42:56.938]
```

- **Board Logs:** Logs relating to the operation of the device (infrastructure) that are non-call session related (e.g., device reset or Web login). These logs are identified by a board ID ("BID"), described in detail in the table below. For example:

```
11:58:30.820 10.15.77.55 local0.notice [S=534370] [BID=50dcb2:31] Activity
Log: WEB: User logout. User: Admin. Session: WEB (10.15.77.100) [Time:10-
09@10:57:16.360]
```

The format of the Syslog message is described in the following table:

Table 52-3: Syslog Message Format Description

Message Item	Description
Timestamp	When the Network Time Protocol (NTP) is enabled, a timestamp string [hour:minutes:seconds.msec] is added to all Syslog messages, for example (in bold): **10:44:11.299** 10.15.77.55 local0.notice [S=511941] [SID=50dcb2:31:12079] (N 483455) ReleaseAddress. IPv4IF=1 IPv6IF=-1 Port=7500 [Time:10-09@09:42:56.938]
IP Address	The device that generated the Syslog message, defined by IP address.
Severity Type	Each Syslog message is generated with a severity level in the format

Message Item	Description
	<*FacilityCode.Severity*>, for example: `10:44:11.299 10.15.77.55 local0.notice [S=511941] [SID=50dcb2:31:12079] (N 483455) ReleaseAddress. IPv4IF=1 IPv6IF=-1 Port=7500 [Time:10-09@09:42:56.938]` The severity level can be one of the following: ■ Error: Indicates that a problem has been identified that requires immediate handling. ■ Warning: Indicates an error that might occur if measures are not taken to prevent it. ■ Notice: Indicates that an unusual event has occurred. ■ Info: Indicates an operational message. ■ Debug: Messages used for debugging. **Note:** ■ The Info and Debug severity-level messages are required only for advanced debugging. By default, they are not sent by the device. ■ Syslog messages displayed in the Web interface (see Viewing Syslog Messages on page 1078) are color coded according to severity level.

Sequence Number [S=<number>]	By default, Syslog messages are sequentially numbered in the format *[S=<number>]*, for example, "[S=538399]". A skip in the number sequence of messages indicates a loss in message packets. The following example of a Syslog shows two missing messages (S=538402 and S=538403):
	`12:11:42.709 10.15.77.55 local0.notice [S=538399] [SID=50dcb2:31:12754] (N 508552) CAC: Remove SBC Outgoing Other, IPG 2 (Teams): 0, SRD 0 (DefaultSRD): 0, SipIF 1 (Teams): 0 [Time:10-09@11:10:28.848]`
	`12:11:42.709 10.15.77.55 local0.notice [S=538400] [SID=50dcb2:31:12754] (N 508553) States: (#2698)SBCCall[Deallocated] [Time:10-09@11:10:28.848]`
	`12:11:42.709 10.15.77.55 local0.notice [S=538401] [SID=50dcb2:31:12754] (N 508554) CAC: Remove SBC Incoming Other, IPG 2 (Teams): 0, SRD 0 (DefaultSRD): 0, SipIF 1 (Teams): 0 [Time:10-09@11:10:28.848]`

Message Item	Description
	`12:11:42.710 10.15.77.55 local0.notice [S=538404] [SID=50dcb2:31:12754] (N 508555) States: (#2699)SBCCall[Deallocated] [Time:10-09@11:10:28.848]` **Note:** To exclude the message sequence number from Syslog messages, configure the 'CDR Syslog Sequence Number' parameter to **Disable** (see Configuring Syslog).
Session ID (SID)	The SID is a unique SIP call session and device identifier. The device identifier facilitates debugging by clearly identifying the specific device that sent the log message, which is especially useful in deployments consisting of multiple devices. In addition, the benefit of unique numbering is that it enables you to filter information (such as SIP, Syslog, and media) according to device or session ID. The syntax of the session and device identifiers is as follows: *[SID=<last 6 characters (3 lower bytes) of MAC address>:<number of times device has reset>:<unique SID counter indicating the call session, which increments consecutively for each new session and resets to 1 after a device reset>]* For example: `10:44:11.299 10.15.77.55 local0.notice [S=511941] [SID=50dcb2:31:12079] (N 483455) ReleaseAddress. IPv4IF=1 IPv6IF=-1 Port=7500 [Time:10-09@09:42:56.938]` Where: ■ *50dcb2* is the device's MAC address. ■ *31* is the number of times the device has reset. ■ *12079* is a unique SID session number (in other words, this is call session 12,079 since the last device reset). ✓ A session includes both the outgoing and incoming legs, where both legs share the same session number. ✓ Forked legs and alternative legs share the same session number.
Board ID (BID)	The BID is a unique non-SIP session related (e.g., device reset) and device identifier. The BID value is similar to the SID (above), except that it doesn't contain the session ID. The device identifier facilitates debugging by clearly identifying the specific device that sent the log message, which is especially useful in deployments consisting of multiple devices. In addition, the benefit of unique numbering is that it

Message Item	Description
	enables you to filter information according to device. The syntax of the BID is as follows: *[BID=<last 6 characters (3 lower bytes) of MAC address>:<number of times device has reset>]* For example: `11:58:30.820 10.15.77.55 local0.notice [S=534370] [BID=50dcb2:31] Activity Log: WEB: User logout. User: Admin. Session: WEB (10.15.77.100) [Time:10-09@10:57:16.360]` Where: ■ *50dcb2* is the device's MAC address. ■ *31* is the number of times the device has reset.
Message Body	Describes the message. For example, the body (shown in bold) of the following Syslog message indicates that the user logged out of the Web interface: `11:58:30.820 10.15.77.55 local0.notice [S=534370] [BID=50dcb2:31] **Activity Log: WEB: User logout. User: Admin. Session: WEB (10.15.77.100)** [Time:10-09@10:57:16.360]`

Event Representation in Syslog Messages

The device denotes events in Syslog message using unique abbreviations, as listed in the following table. For example, if an invalid payload length event occurs, the Syslog message uses the abbreviated event string "IP":

```
Apr  4 12:00:12 172.30.1.14 IP:5  [Code:0x5004] [CID:3294] [Time: 20:17:00]
```

> For Syslog messages sent for packet loss events, see Packet Loss Indication in Syslog on page 1083.

Table 52-4: Syslog Error Event Abbreviations

Error Abbreviation	Error Name Description
AA	Invalid Accumulated Packets Counter
AC	Invalid Channel ID

Error Abbreviation	Error Name Description
AL	Invalid Header Length
AO	Invalid Codec Type
AP	Unknown Aggregation Payload Type
AR	Invalid Routing Flag Received
AT	Simple Aggregation Packets Lost
CC	Command Checksum Error
CE	Invalid Cell Coder Code
CS	Command Sequence Error
ES	8 sec Timeout Before Disconnect
HO	Host Received Overrun
IA	Invalid AMR Payload
IC	Invalid CID Error
IG	Invalid G723 Code
IP	Invalid payload length
IR	Invalid RTCP Packet
IS	Invalid SID Length
LC	Transmitter Received Illegal Command
LF	Lost Fax Frames In High Speed Mode
LM	Lost Modem Frames In High Speed Mode
MI	Misalignment Error
MR	Modem Relay Is Not Supported
PD	RTP Packet Duplicated
OR	DSP JB Overrun
PH	Packet Header Error

Error Abbreviation	Error Name Description
RB	Counts the number of BFI Frames Received From The Host
RD	No Available Release Descriptor
RO	RTP Reorder
RP	Unknown RTP Payload Type
RS	RTP SSRC Error
UF	Unrecognized Fax Relay Command

Syslog Fields for Answering Machine Detection (AMD)

The Syslog message can include information relating to the Answering Machine Detection (AMD) feature. AMD is used to detect whether a human (including a fax machine), an answering machine, silence, or answering machine beeps have answered the call on the remote side.

- AMDSignal – the field can acquire one of the following values:
  - voice (V)
  - answer machine (A)
  - silence (S)
  - unknown (U)
- AMDDecisionProbability – probability (in %) success that correctly detects answering type

Below is an example of such a Syslog message with AMD information:

```
CallMachine:EVENT_DETECTED_EV - AMDSignal = <type - V/A/S/U>,
AMDDecisionProbability = <percentage> %
```

If there is no AMD detection, the AMDSignal field is shown empty (i.e. AMDSignal =).

For more information on the AMD feature, see Answering Machine Detection (AMD).

SNMP Alarms in Syslog Messages

SNMP alerts are sent to the Syslog server using the following formats:

- **Raised Alarms:** RAISE-ALARM: <Alarm Name>; Textual Description: <Textual Description>; Severity <Alarm Severity>; Source <Alarm Source>; Unique ID: <Alarm Unique ID >.

  If additional information exists in the alarm, then these are also added: Additional Info1:/ Additional Info2:/ Additional Info3

  The Messages' Severity is as follows:

Table 52-5: Syslog Message Severity

ITU Perceived Severity (SNMP Alarm's Severity)	AudioCodes Syslog Severity
Critical	RecoverableMsg
Major	RecoverableMsg
Minor	RecoverableMsg
Warning	Notice
Indeterminate	Notice
Cleared	Notice

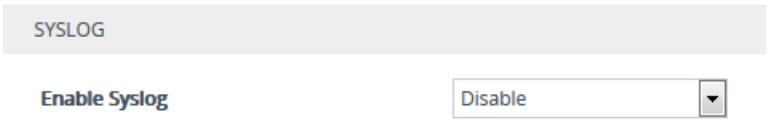
- **Cleared Alarms:** CLEAR-ALARM: <Alarm Name>; Textual Description: <Textual Description>; Severity <Alarm Severity>; Source <Alarm Source>; Unique ID: <Alarm Unique ID >; If exists Additional Info1:/ Additional Info2:/ Additional Info3:

Enabling Syslog

To use Syslog, you first need to enable it.

➢ **To enable Syslog:**

1. Open the Logging Settings page (**Troubleshoot** menu > **Troubleshoot** tab > **Logging** folder > **Logging Settings**).
2. From the 'Enable Syslog' drop-down list, select **Enable**.



3. Click **Apply**.

Configuring the Syslog Server Address

The device sends the Syslog messages to the Syslog server's address.

- The Syslog IP address configuration described in this section is also used for the CDR server, unless you have configured a dedicated address for the CDR server (see Enabling CDR Generation and Configuring CDR Server Address on page 985).
- The Syslog port number configuration described in this section also applies to the CDR server.

➢ **To configure the Syslog server address:**

1. Open the Logging Settings page (**Troubleshoot** menu > **Troubleshoot** tab > **Logging** folder > **Logging Settings**).
2. In the 'Syslog Server IP' field [SyslogServerIP], enter the IP address of the Syslog server.
3. In the 'Syslog Server Port' field, enter the port of the Syslog server.

4. Click **Apply**.

Configuring Syslog Message Severity Level

You can configure the minimum severity level of messages that you want to include in Syslog messages that are generated by the device.

It's **strongly recommended** to leave the Syslog severity level at its default setting. Changing severity level is typically done only by AudioCodes Support for debugging.

The severity levels are described in the following table.

Severity Level (Highest to Lowest)	Syslog String	Description
Fatal	emerg	A panic condition (system is unstable).
Alert	alert	A problem has been identified and an action must be taken immediately.
Critical	crit	A problem has been identified that is critical.
Error	error	An error has been identified.
Warning	warning	An error that might occur if measures are not taken to prevent it.
Notice	notice	An unusual event has occurred.
Informational	info	An operational message.
Debug	debug	Debug message.

The specified severity level and all higher severity levels are included in the Syslog message. For example, if you configure the parameter to **Alert**, the Syslog includes messages with **Alert** severity level and messages with **Fatal** severity level.

When viewing Syslog messages in the Web interface (see Viewing Syslog Messages on page 1078), each severity level is displayed in a different color.

➢ **To configure the minimum message severity level to include in Syslog:**

1. Open the Logging Settings page (**Troubleshoot** menu > **Troubleshoot** tab > **Logging** folder > **Logging Settings**).
2. From the 'Log Severity Level' [SyslogLogLevel] drop-down list, select the severity level.

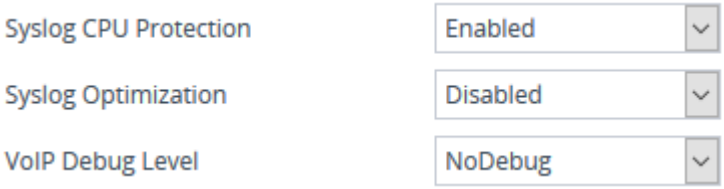
3. Click **Apply**.

Configuring Syslog Debug Level

You can configure the amount of information (debug level) to include in Syslog messages. You can also enable the device to send multiple Syslog messages bundled into a single packet, and enable a protection mechanism that automatically lowers the debug level when the device's CPU resources become low, ensuring sufficient CPU resources are available for processing voice traffic.

➢ **To configure the Syslog debug level:**

1. Open the Logging Settings page (**Troubleshoot** menu > **Troubleshoot** tab > **Logging** folder > **Logging Settings**).



2. From the 'VoIP Debug Level' [GwDebugLevel] drop-down list, select the debug level of Syslog messages:
   - **No Debug:** Disables Syslog and no Syslog messages are sent.
   - **Basic:** Sends debug logs of incoming and outgoing SIP messages.
   - **Detailed:** Sends debug logs of incoming and outgoing SIP message as well as many other logged processes.
3. From the 'Syslog Optimization' [SyslogOptimization] drop-down list, select whether you want the device to accumulate and bundle multiple debug messages into a single UDP packet before sending it to a Syslog server. The benefit of the feature is that it reduces the

number of UDP Syslog packets, thereby improving (optimizing) CPU utilization. The size of the bundled message is configured by the [MaxBundleSyslogLength] parameter.

4. From the 'Syslog CPU Protection' [SyslogCpuProtection] drop-down list, select whether you want to enable the protection feature for the device's CPU resources during debug reporting, ensuring voice traffic is unaffected. If CPU resources drop (i.e., high CPU usage) to a critical level (user-defined threshold), the device automatically lowers the debug level to free up CPU resources that were required for the previous debug-level functionality. When CPU resources become available again, the device increases the debug level to its' previous setting. For example, if you set the 'Debug Level' to **Detailed** and CPU resources decrease to the defined threshold, the device automatically changes the level to **Basic**, and if that is not enough, it changes the level to **No Debug**. Once CPU resources are returned to normal, the device automatically changes the debug level back to its' original setting (i.e., **Detailed**). The threshold is configured by the [DebugLevelHighThreshold] parameter.
5. Click **Apply**.

Reporting Management User Activities

The device can report operations (activities) performed in the device's management interfaces (e.g., Web and CLI) by management users, in Syslog messages. The Syslog message indicates these logs with the string "Activity Log". Each logged user activity includes the following information:

- Username (e.g., "Admin") of the user that performed the action
- IP address of the client PC from where the Web user accessed the management interface
- Protocol used for the session (e.g., SSH or HTTP)

The following example shows a Web-user activity log (indicating a login action) with the above-mentioned information:

```
14:07:46.300 : 10.15.7.95 : Local 0   :NOTICE  : [S=3149] [BID=3aad56:32]
Activity Log: WEB: Successful login at 10.15.7.95:80. User: Admin. Session:
HTTP (10.13.22.54)
```

The device can report the following user activities:

- Modifications of individual parameters, for example:

```
14:33:00.162 : 10.15.7.95 : Local 0   :NOTICE  : [S=3403] [BID=3aad56:32]
Activity Log: Max Login Attempts was changed from '3' to '2'. User: Admin.
Session: HTTP (10.13.22.54)
```

- Modifications of table fields, and addition and deletion of table rows, for example:

14:42:48.334 : 10.15.7.95 : NOTICE : [S=3546] [BID=3aad56:32] Activity Log: **Classification - remove line 2.** User: Admin. Session: HTTP (10.13.22.54)

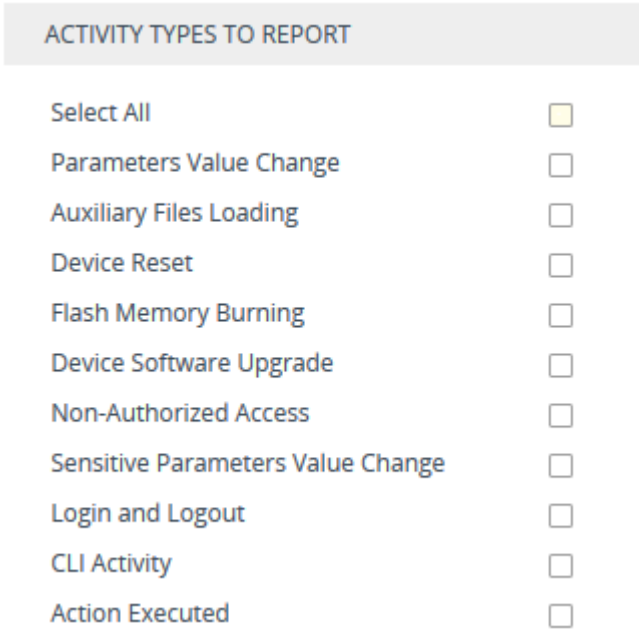
- Entered CLI commands (modifications of security-sensitive commands are logged without the entered value).
- Configuration file load (reported without per-parameter notifications).
- Auxiliary file load and software update.
- Device reset and burn to flash memory.
- Access to unauthorized Web pages according to the Web user's access level.
- Modifications of "sensitive" parameters.
- Log in and log out.
- Actions not related to parameter changes (for example, file uploads, file delete, lock-unlock maintenance actions, LDAP clear cache, register-unregister, and start-stop trunk). In the Web, these actions are typically done by clicking a button (e.g., the LOCK button).

For more information on each of the above listed options, see Syslog, CDR and Debug Parameters.

The following procedure describes how to configure management user activity logging through the Web interface. You can also configure it through ini file [ActivityListToLog] or CLI (`configure troubleshoot > activity-log`).

➢ **To configure reporting of management user activities:**

1. Open the Logging Settings page (**Troubleshoot** tab > **Troubleshoot** menu > **Logging** folder > **Logging Settings**).
2. Under the Activity Types to Report group, select the actions to report to the Syslog server. To select (or deselect) all activity types, click the 'Select All' check box.



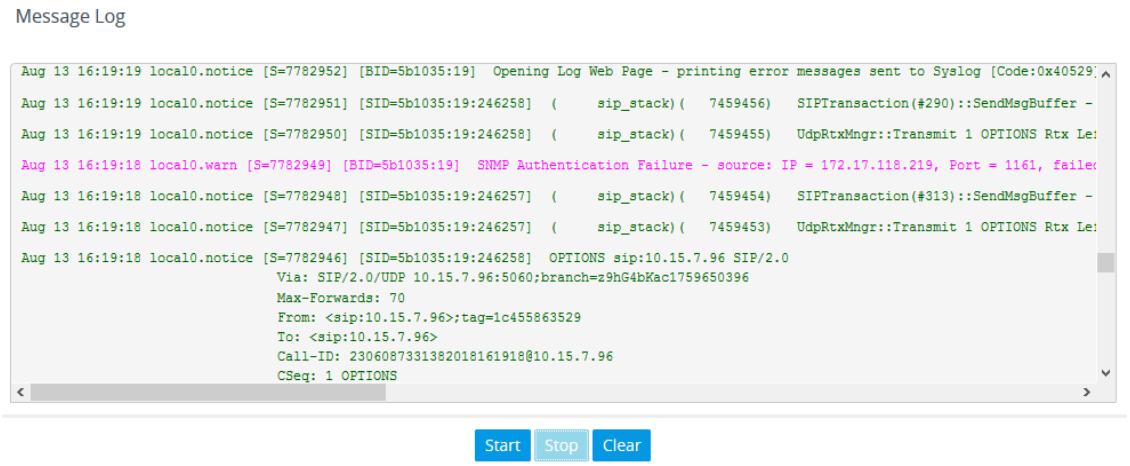
3. Click **Apply**.

- You can also view logged user activities in the Web interface (see Viewing Web User Activity Logs).
- Logging of CLI commands can only be configured through CLI or ini file.
- You can configure the device to send an SNMP trap each time a user performs an action. For more information, see Enabling SNMP Traps for Web Activity on page 84.

Viewing Syslog Messages

You can view Syslog messages generated by the device using any of the following Syslog server types:

- **Device's Web Interface:** The device provides an embedded Syslog server, which is accessed through the Web interface (**Troubleshoot** tab > **Troubleshoot** menu > **Message Log**). You can select the Syslog messages displayed on the page, and then copy-and-paste them into a text editor such as Notepad. This text file (*txt*) can then be sent to AudioCodes support team for diagnosis and troubleshooting.



The displayed logged messages are color-coded based on message type:

- **"notice":** Dark green
- **"error", "crit", "alert", "emerg":** Red
- **"debug":** Black
- **"info":** Blue
- **"warn":** Magenta

The page provides various buttons to do the following actions:

Table 52-6: Buttons on Message Log Page

Button	Description
Start	Resumes the message log after it has been stopped (see the **Stop** button).
Stop	Stops the message log, allowing you to easily scroll through the messages to a specific message.
Clear	Clears the message log. The button can only be clicked after you have stopped the message log (see the **Stop** button). **Note:** If you navigate away from the Message Log page to another page, the Message Log is stopped and cleared.

- It's not recommended to keep a Message Log session open for a prolonged period. This may cause the device to overload. For prolonged (and detailed) debugging, use an external Syslog server.
- The Message Log page provides limited Syslog server functionality.

- **Device's CLI:** The device sends error messages (e.g., Syslog messages) to the CLI as well as to the configured destination.
  - To start debug recording:

```
debug log
```

- To stop debug recording:

```
no debug log
```

- To stop all debug recording:

```
no debug log all
```

- **Wireshark:** Third-party, network protocol analyzer (http://www.wireshark.org).

> When debug recording is enabled and Syslog messages are also included in the debug recording, to view Syslog messages using Wireshark, you must install AudioCodes' Wireshark plug-in (acsyslog.dll). Once the plug-in is installed, Syslog messages are decoded as "AC SYSLOG" and displayed using the "acsyslog" filter (instead of the regular "syslog" filter). For more information on debug recording, see Debug Recording.

- **AudioCodes Syslog Viewer:** This utility can be used for two major tasks:
  - Recording and displaying syslog messages from the device
  - Analyzing recorded logs (including support for interactive SIP ladder diagrams)

  To obtain the Syslog Viewer installation file, download it from https://www.audiocodes.com/library/firmware.

Figure 52-1: Example of Syslog Messages in Syslog Viewer

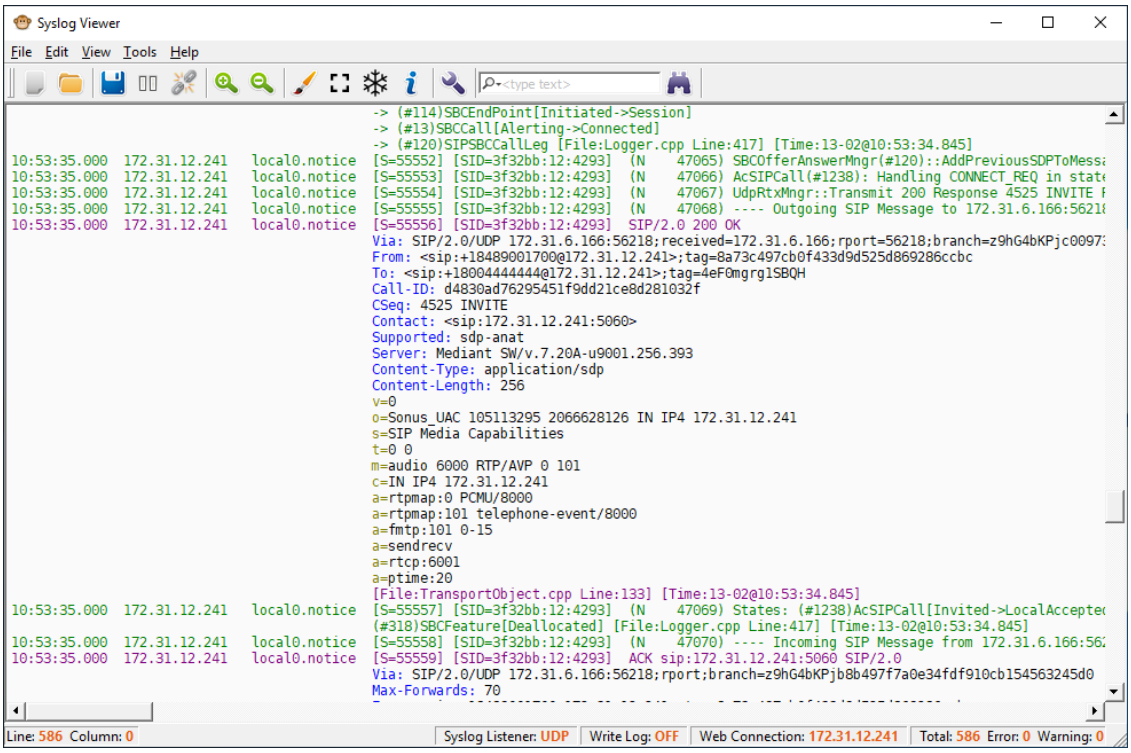
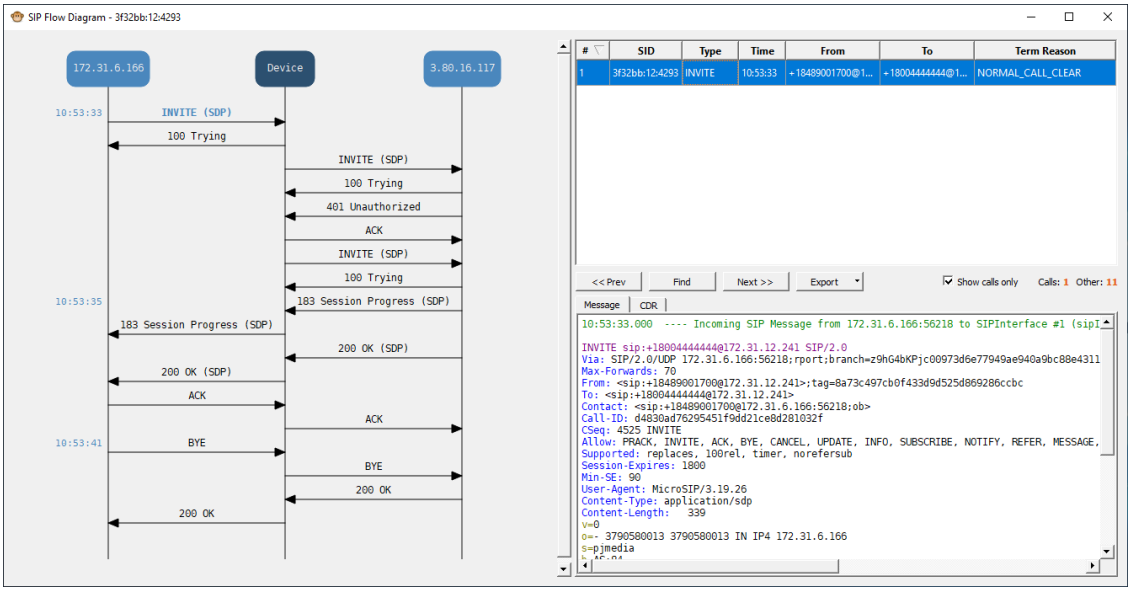


Figure 52-2: Example of SIP Ladder Diagram in Syslog Viewer



- **Third-party, Syslog Server:** Any third-party, Syslog server program that enables filtering of messages according to parameters such as priority, IP sender address, time, and date.

Syslog Message Description for CPU Overload

Whenever the device detects a CPU overload, it sends a Syslog message that shows CPU utilization of the different processes (tasks) per core. This information can help in identifying the cause of the overload. When the devicedetects a CPU overload, it sends a Syslog message every 10 seconds until it returns to normal state.

You can also view CPU utilization through the CLI, by using the following command:
`show system utilization`

The figure below shows an example of a Syslog message generated because of a CPU overload. CPU utilization information is shown under the "CPUUtilMonitor" section (shown in pink). The subsequent table describes the displayed information.

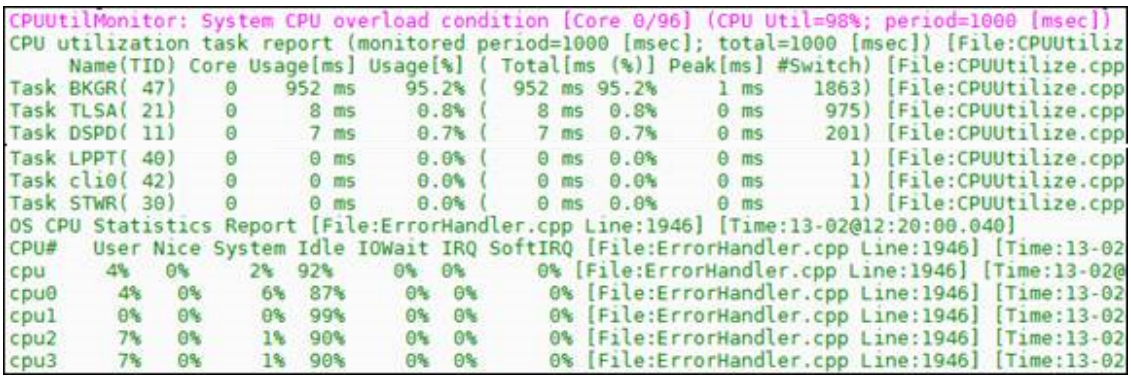


Table 52-7: CPU Overload Fields Description in Syslog Message

Field	Description
First line (shown in pink)	
“Core”	Index of the CPU core.
“CPU Util”	CPU utilization (in percentage).
“period”	Total period (in msec).
Second line	
“monitored period”	Duration (in msec) of CPU overload within the total monitored period.
“total“	Monitored period (in msec).
Statistics per task (process) in overloaded cores only **Note:** By default, the Syslog message only shows the five most used tasks in the last period.	
“Name (TID)”	Name of task (process).
“Core”	Index of the CPU core.
“Usage [ms]”	Total time (msec) of monitored period that the task utilized CPU.
“Usage [%]”	Percentage of time of monitored period that the task utilized

Field	Description
	CPU.
“Total [ms (%)]”	Total time (in msec) and percentage that task utilized CPU during entire period.
“peak [ms]”	Maximum lasting time (msec) that the task utilized CPU during the period.
“#Switch”	Context switch time - number of consecutive periods that were allocated for this task.
Statistics per CPU core	
"CPU#"	Index of the CPU core.
"User"	Percentage of CPU utilization that occurred while executing at the user level (application).
"Nice"	Percentage of CPU utilization that occurred while executing at the user level with nice priority (Linux systems).
"System"	Percentage of CPU utilization that occurred while executing at the system level (kernel).
"Idle"	Percentage of time that the CPU was idle (%) during which no tasks were using the CPU core.
"IOWait"	Percentage of time that the CPU was idle (5) during which tasks were using the CPU core.
"IRQ"	IRQ time (in percentage).
"SoftIRQ"	SoftIRQ time (in percentage%).

Packet Loss Indication in Syslog

The device reports packet loss (PL) of incoming (Rx) RTP media streams (calls) in 15-second intervals. The device obtains packet loss statistics from the RTCP of the RTP streams. When packet loss occurs in the 15-second interval, at the end of the interval the device sends a Syslog message with Warning severity level, indicating this packet loss. The Syslog indicates the number of calls that experienced packet loss per packet loss range (in percentage) during the interval. It also indicates the number of calls that didn't have packet loss. If no packet loss occurred in all the RTP streams in the 15-second interval, no Syslog message is sent.

Below shows an example of a Syslog message sent when packet loss occurred in the 15-second interval. This Syslog indicates that 6 calls were active during the interval. One call had no packet loss, 3 calls had 1 to 2% packet loss, and 2 calls had 5 to 100% packet loss:

```
16:47:13.921 192.168.8.70 local0.warn [S=2116] [BID=884772:92] Packets-Loss report [PL range]=#media-legs: [No PL]=1, [up to 0.5%]=0, [0.5% - 1%]=0, [1% - 2%]=3, [2% - 5%]=0, [5% - 100%]=2 [[Time:28-12@00:40:18.550|time:28-12@00:40:18.550]]
```

Below shows the default packet-loss ranges in the Syslog:

- [No PL]: Indicates the number of calls without packet loss.
- [up to 0.5%]: Indicates the number of calls with up to 0.5% packet loss. This packet loss typically has no effect on voice quality.
- [0.5% - 1%]: Indicates the number of calls with 0.5 to 1% packet loss. This packet loss typically has no effect on voice quality.
- [1% - 2%]: Indicates the number of calls with 1 to 2% packet loss. This packet loss may affect voice quality for calls using certain vocoders.
- [2% - 5%]: Indicates the number of calls with 2 to 5% packet loss. This packet loss affects voice quality and typically indicates a network problem.
- [5% - 100%]: Indicates the number of calls with 5 to 100% packet loss. This packet loss affects voice quality and typically indicates a network problem.

You can change these packet-loss ranges, using the [PLThresholdLevelsPerMille] parameter. For more information, see Syslog, CDR and Debug Parameters on page 1176.

- The packet loss report in the Syslog message should be carefully considered. For example, for calls that are opened and then closed during the 15-second interval, packet loss statistics may be misleading due to insufficient packets for accurate calculation. Therefore, if the Syslog message shows very few calls in the high packet-loss ranges, then you should probably ignore them as it might be due to this scenario. On the other hand, if there is a large number of calls falling into these high packet-loss ranges, then it probably indicates network problems.

Configuring Debug Recording

This section describes how to configure debug recording and how to collect debug recording packets.

- If debug recording is sent to a debug recording server (see Configuring the Debug Recording Server Address below), the device's OAMP interface is used by default.
- For a detailed description of the debug recording parameters, see Syslog, CDR and Debug Parameters.

Configuring the Debug Recording Server Address

The procedure below describes how to configure the address of the debug recording server to where the device sends the captured traffic. Once you configure an address, the device generates debug recording packets for all calls. However, you can configure the device to generate debug recording packets for specific calls, using Logging Filter rules in the Logging Filters table (see Configuring Log Filter Rules).

- When the debug recording server is configured with an IPv4 address, the device sends the debug recording packets through its OAMP interface, by default.

➢ **To configure the debug recording server's address:**

1. Open the Logging Settings page (**Troubleshoot** tab > **Troubleshoot** menu > **Logging** folder > **Logging Settings**).
2. In the 'Debug Recording Destination IP' field, configure the IP address (IPv4) of the debug capturing server.
3. In the 'Debug Recording Destination Port' field, configure the port of the debug capturing server.
4. Click **Apply**.

Collecting Debug Recording Messages

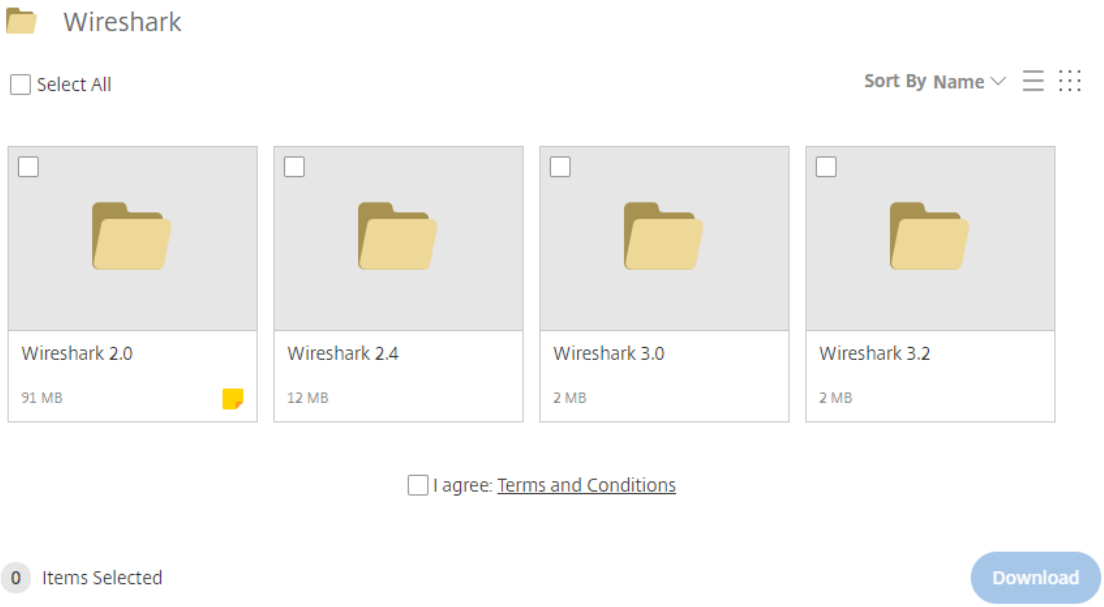
To collect debug recording packets, use the open source packet capturing program, Wireshark. To analyze AudioCodes debug recording protocol, proprietary Wireshark plug-in files are required.

- The default debug recording port is 925. You can change the port in Wireshark (**Edit** menu > **Preferences** > **Protocols** > **AC DR**).
- The plug-in files are per major software release of Wireshark. For more information, contact the sales representative of your purchased device.
- Make sure that you download the plug-in files that match your computer's Windows operating system (32-bit or 64-bit processor).
- The source IP address of the messages is always the OAMP IP address of the device.

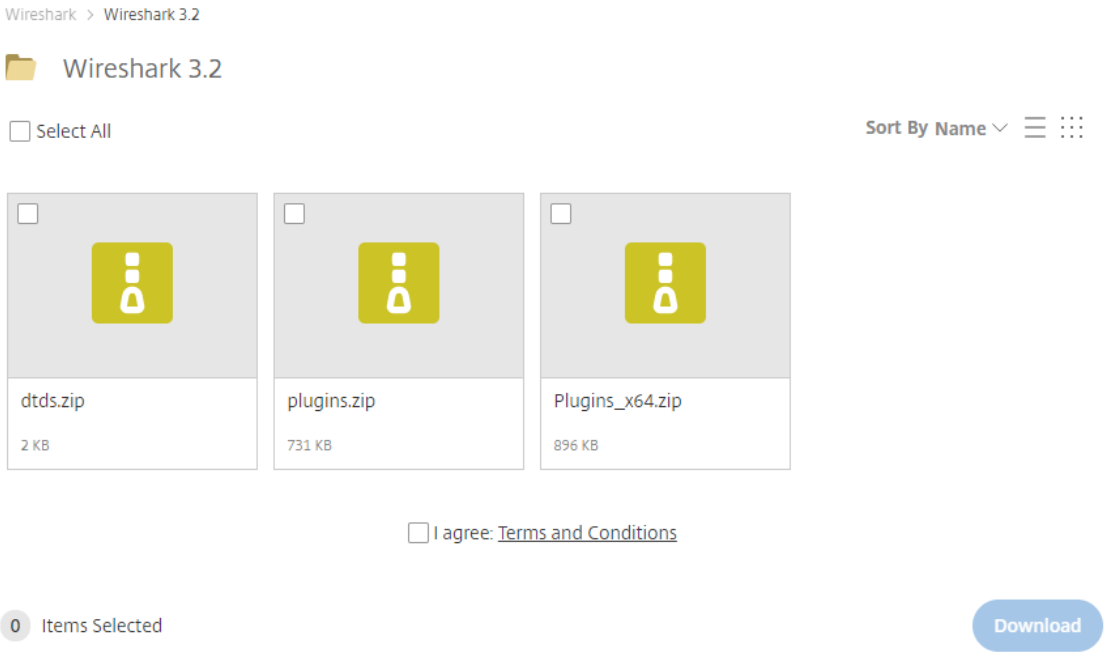
➢ **To view debug recording messages using Wireshark:**

1. Install Wireshark on your computer (which can be downloaded from https://www.wireshark.org).
2. Download AudioCodes proprietary Wireshark plug-in files according to the type of installation (32-bit or 64-bit):
   a. Got to AudioCodes firmware download website page at https://www.audiocodes.com/library/firmware, and then navigate to the page for the "Wireshark Plugins":

   b. Click the area shown above; folders containing the plug-in files for different Wireshark versions are displayed, as shown in the example below:



c. Click the folder icon of the required Wireshark version; zipped folders of the selected Wireshark version are displayed:



d. Select the check box of the required zipped plug-in files, select the **I agree** (to the terms and conditions) check box, and then click **Download**; the zipped folder is downloaded to your computer.

Make sure that you select the zipped plug-in folder that matches your computer's Windows operating system (32-bit or 64-bit processor):

- **plugins.zip:** For 32-bit
- **Plugins_x64.zip:** for 64-bit

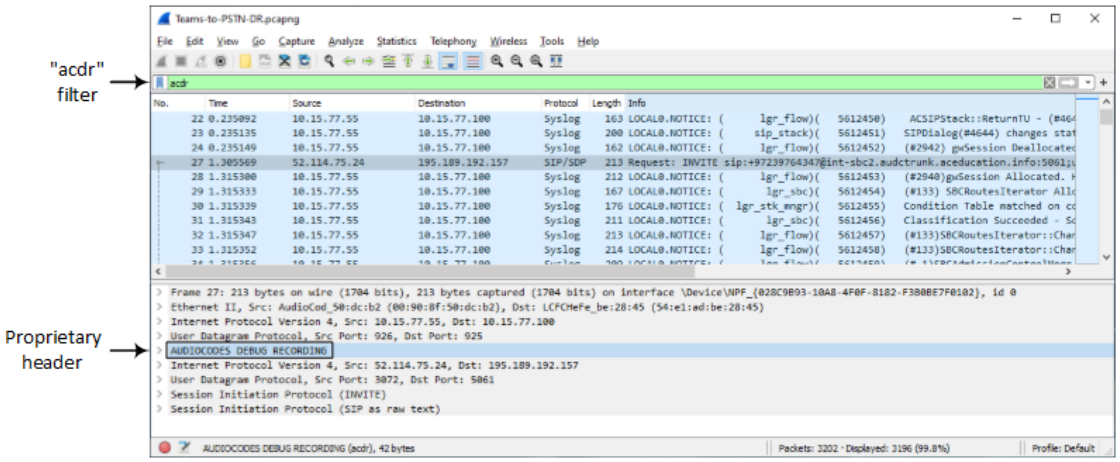
e. Unzip the downloaded plug-in folder; a folder containing all the plug-in files (.dll) is created.

f. Copy all the .dll files to the *plugin* folder (or for Wireshark Version 3.0 or later, to *plugins\<Wireshark version>\epan*) of the Wireshark installation. If the folder already has existing .dll files with the same name, overwrite them.

3. Start your Wireshark program.

4. In the filter field, type "acdr" to view the debug recording messages.

The device adds the header "AUDIOCODES DEBUG RECORDING" to each debug recording message, as shown below:



Debug Capturing on Physical VoIP Interfaces

You can capture traffic on the device's physical (Ethernet LAN) VoIP interfaces (Layer-2 VLAN tagged packets). The captured traffic can be saved in a PCAP-format file (suitable for Wireshark) to a TFTP (default) or an FTP server. The generated PCAP file is in the Extensible Record Format (ERF). The maximum file size of debug captures that can be saved to the device is 20 MB100 MB.

To capture traffic on physical VoIP interfaces, use the following CLI commands:

- Starts physical VoIP debug capture:

```
# debug capture voip physical eth-lan
# debug capture voip physical start
```

- Captures packets continuously in a cyclical buffer (packets always captured until stop command):

```
# debug capture VoIP physical cyclic buffer
```

- Retrieves latest capture (PCAP file) saved on a specified server:

```
# debug capture VoIP physical get_last_capture <TFTP/FTP server IP address>
```

  The file is saved to the device's memory (not flash) and erased after a device reset.

- Marks the captured file (useful for troubleshooting process):

```
# debug capture VoIP physical insert-pad
```

  Before running this command, the debug capture must be started.

- Displays debug status and configured rules:

```
# debug capture VoIP physical show
```

- Specifies the destination (FTP, TFTP, or USB) where you want the PCAP file sent:

```
# debug capture VoIP physical target <ftp|tftp|usb>
```

- Stops the debug capture, creates a file named debug-capture-voip-<timestamp>.pcap, and sends it to the TFTP or FTP server:

```
# debug capture voip physical stop <TFTP/FTP server IP address>
```

If no IP address is defined, the capture is saved on the device for later retrieval.

Debug Capturing on VoIP Interfaces

You can capture network traffic on the device's VoIP interfaces per VLAN device that is configured in the Ethernet Devices table (see Configuring Underlying Ethernet Devices on page 120). You can send the captured traffic to the following:

- CLI terminal screen (tcpdump format): The captured network packets are displayed in the CLI until you end the capture by pressing the CTRL + C key combination.
- Remote server (TFTP or FTP): The capture is saved as a PCAP file (suitable for Wireshark) and sent to a specified server (default is TFTP) . The generated PCAP file is in the Extensible Record Format (ERF) and is saved on the device during the capture. The maximum file size that can be saved to the device is 10 MB and as long as the capture continues, the packets are written to this 10-MB file in a cyclic manner. When you end the capture (by pressing the CTRL + C key-combination), the device sends the capture file to the server.

➢ **To capture traffic on a VLAN VoIP device:**

1. Define the VLAN ID on which you want to do the capture:

```
# debug capture voip interface vlan <VLAN ID>
```

2. Define the protocol that you want to capture (all|arp|icmp|ip|ipv6|tcp|udp):

```
# debug capture voip interface vlan <VLAN ID> proto <Protocol>
```

3. Define a source and/or destination IP address to be captured (any|ipv4_address|ipv6_address):

```
# debug capture voip interface vlan <VLAN ID> proto <Protocol> host <IP Address>
```

At this stage, you can press Enter to output the capture to the CLI terminal window, or you can continue with the next step to configure additional commands.

4. Define a source and/or destination port number to be captured (any|[1-65535]):

```
# debug capture voip interface vlan <VLAN ID> proto <Protocol> host <IP Address> port <Port>
```

At this stage, you can press Enter to output the capture to the CLI terminal window, or you can continue with the next step to configure additional commands.

5. Define the IP address (IPv4) of the server (TFTP or FTP) to where you want the device to send the captured file:

```
# debug capture voip interface vlan <VLAN ID> proto <Protocol> host <IP Address> ftp-server|tftp-server <IP Address>
```

6. Press Enter to save the capture to a file on the device.
7. Press the CTRL + C key-combination to stop the capture and to send the file to the defined server.

53 Creating Core Dump and Debug Files upon Device Crash

For debugging, you can configure the device to create a core dump file and a debug file. These files may assist you in identifying the cause of the crash. The core dump can either be included in or excluded from the debug file, or alternatively, sent separately to a TFTP server. You can then provide the files to AudioCodes support team for troubleshooting.

Enabling Core Dump File Generation

You can enable the device to generate a core dump file upon a device crash. The core dump is a copy of the memory image at the time of the crash. Each time the device crashes, it creates a new core dump file, which replaces the previous core dump file (if exists). The core dump file provides a powerful tool for determining the root cause of the crash.

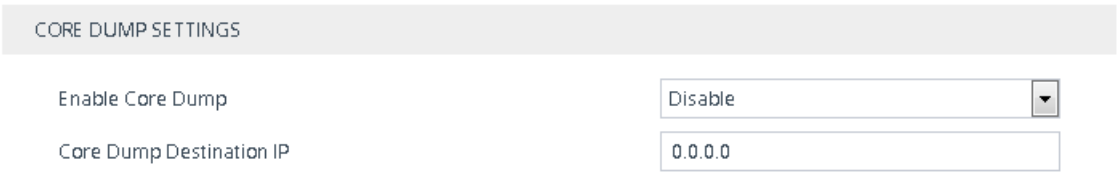
You can configure the device to send the core dump file to a TFTP server (defined by an IP address). If you don't configure an address, the core dump file is saved on the device's flash memory (if it has sufficient memory).

The core dump file is saved as a binary file, using the following file name format: *core_<Device Name>_ver_<Firmware Version>_mac_<MAC Address>_<Date>_<Time>*. For example, "core_acDevice_ver_720-8-4_mac_00908F099096_1-02-2015_3-29-29".

> **Note:** When downloading the debug file to your computer, you can also include the core dump file, as described in Downloading the Debug (and Core Dump) File on the next page.

➢ **To enable core dump file generation:**

1. (Optional) Set up a TFTP server to where you want the device to send the core dump file.
2. Open the Debug Files page (**Troubleshoot** menu > **Troubleshoot** tab > **Debug** folder > **Debug Files**).



3. From the 'Enable Core Dump' drop-down list, select **Enable**.
4. (Optional) If you want the device to send the core dump file to a remote TFTP server, then in the 'Core Dump Destination IP' field, enter the IP address of the remote server. If not configured, the device saves the file on its storage memory.
5. Click **Apply**, and then reset the device with a save-to-flash for your settings to take effect.

Downloading the Debug (and Core Dump) File

You can download the debug file from the device (flash memory) and save it to a folder on your local computer. The device creates the debug file whenever an exception occurs. Each time the device creates a new debug file, it overwrites the existing file.

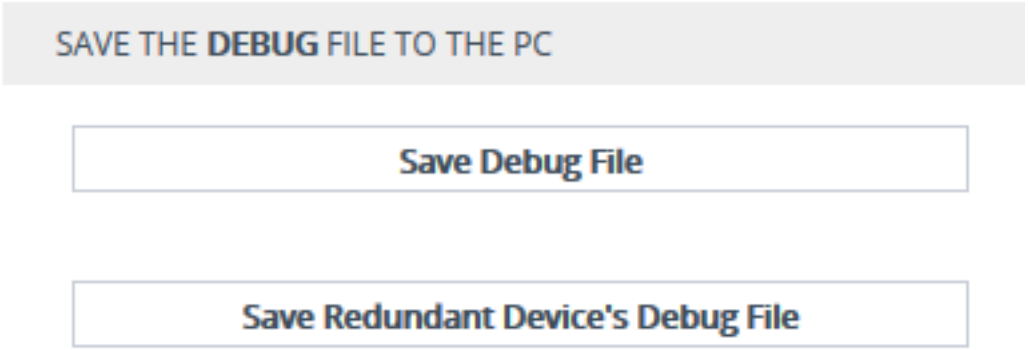
The debug file is saved as a .tar file with the following filename format: *debug_<Device Name>_ver_<Firmware Version>_mac_<MAC Address>_<Date>_<Time>.* For example, "debug_acDevice_ver_720-8-4_mac_00908F099096_1-03-2015_3-29-29.tar".

The debug file contains the following:

- Exception information, indicating the specific point in the code where the crash occurred and a list of up to 50 of the most recent SNMP alarms that were raised by the device before it crashed.
- Reset counter summary file (reset-table-of-content.txt), displaying each reset with the reason (intentional) or exception for the reset, date and time that the reset occurred, and the device's software version.
- Latest syslog messages that were recorded prior to the crash.
- Core dump file, only if **all** of the following conditions are met:
  - You have enabled core dump generation (see Enabling Core Dump File Generation on the previous page).
  - You have not configured an IP address to send the core dump to a remote server (see Enabling Core Dump File Generation on the previous page).
  - The device has sufficient memory on its flash memory.
  - You have enabled the inclusion of the core dump file in the debug file (see below procedure).
- The debug file may include additional application-proprietary debug information.

➢ **To download the debug file:**

1. Open the Debug Files page (**Troubleshoot** menu > **Troubleshoot** tab > **Debug** folder > **Debug Files**), and then scroll down to the 'Save The Debug File To The PC' group:



2. By default, the core dump file is included in the downloaded debug file. If you don't want to include it, clear the 'Attach Core Dump File' check box.
3. Click the **Save Debug File** button; the device downloads the debug file to a folder on your computer. For devices in HA mode, this button downloads the debug file of the active device. To download the redundant device's debug file, click the **Save Redundant Device's Debug File** button.

- Downloading the debug file may take a few minutes. Depending on file size, it may even take more than 10 minutes.
- If the device is operating in the FIPS security mode (see Configuring FIPS Security Mode), download of the core dump file is blocked. For enhanced troubleshooting when operating in FIPS mode, please contact your AudioCodes sales representative.
- You can also download (Get) the debug file from the device through SFTP. The file is located in the device's **/debug** folder. Your SFTP client needs to authenticate itself with the SFTP server (i.e., the device) and access is granted only to users with Security Administrator level. In addition, you must enable SSH on the device.
- The device may take a long time to prepare the debug file for SFTP transfer if it contains much information. Some SFTP clients (for example, WinSCP and FileZilla) have a short default connection timeout and if the file transfer is not started within this timeout, the transfer attempt is aborted. Therefore, it is recommended to configure a longer timeout for your SFTP client application.

Deleting the Debug (and Core Dump) File

You can delete the debug file that is stored on the device's memory. If you have enabled core dump file generation (see Enabling Core Dump File Generation on page 1091) without configuring an address of a remote server to send the file to, the core dump file, which is included with the stored debug file is also deleted.

➢ **To delete the debug file (and core dump file):**

1. Establish a CLI session with the device.
2. Type the following command, and then press Enter:

```
# clear debug-file
```

Viewing Debug (and Core Dump) File Contents

You can view the contents of the downloaded or locally stored debug and core dump files.

- **Downloaded file:** Unzip the downloaded debug file or core dump file. The unzipped file includes the following subfolders:
  - **Device:** This folder contains the following file:

- **configuration-package.tar.gz:** This is the Configuration Package file, as described in Saving and Loading a Configuration Package File on page 898.

- **reset-history:** This folder contains logged device resets and contains the following:

  The *reset-table-of-content.txt* file lists the latest logged device resets, where each logged reset is sequentially numbered ("Counter"), providing the reset reason and the time and date when it occurred. If the reset was caused by an error (i.e., crash), "Exception" (instead of "Reset") is displayed above the reset counter. Below shows an example of logged device resets:

```
** Current Reset Counter [68] **

***** Reset    *****
Reset Counter:67
Reset Reason: Web Reset
Reset Time: 8.9.2020 20.29.13
**************

***** Exception *****
Reset Counter:66
Exception Reason: Linux Signal
EXCEPTION TIME : 8.9.2020 20.15.43
**************

***** Exception *****
Reset Counter:65
Exception Reason: System crashed due to Kernel Panic
EXCEPTION TIME : 31.8.2020 10.16.45
**************
```

  Each logged device reset that is listed in the *reset-table-of-content.txt* file has a subfolder whose name is the reset counter (e.g., "67"). This subfolder contains system events or messages that were logged just prior to the device reset:

  - **core.lzma:** This file is generated If Core Dump is enabled and the device resets (crashes) due to exception event. It is only present in the folder of the latest reset due to an exception.
  - **ExceptionInfo.txt:** This file is generated only if the device reset was caused by an exception event (error). As mentioned previously, these logged device resets are displayed in the *reset-table-of-content.txt* file with the title "Exception". The file contains detailed information of the exception.
  - **ini.lzma:** This file contains the configuration of the device and is generated only when the device crashes (resets due to Exception)
  - **NoSip.lzma:** This file contains the latest Syslog messages, but without SIP-related Syslog messages.

- **Syslog.lzma:** This file contains all the latest Syslog messages.
- **system-logs.lzma:** This folder contains the persistent log files (PersistLogs.tar)

- **CLI:** To view the debug file in CLI, use the following commands:
  - Reset history (list of resets or a specific reset counter): `show debug-file reset-info {list|reset-counter}`
  - Generated file contents (list of files or a specific file): `show debug-file device-logs list|file`

The Core Dump file cannot be viewed in CLI.

54 Debugging Web Services

If you have configured remote Web services (see Remote Web Services), you can enable debugging of the remote HTTP clients. You can configure the debug level from 1 to 3, where 3 is the most detailed. The debug messages are sent to the Syslog server.

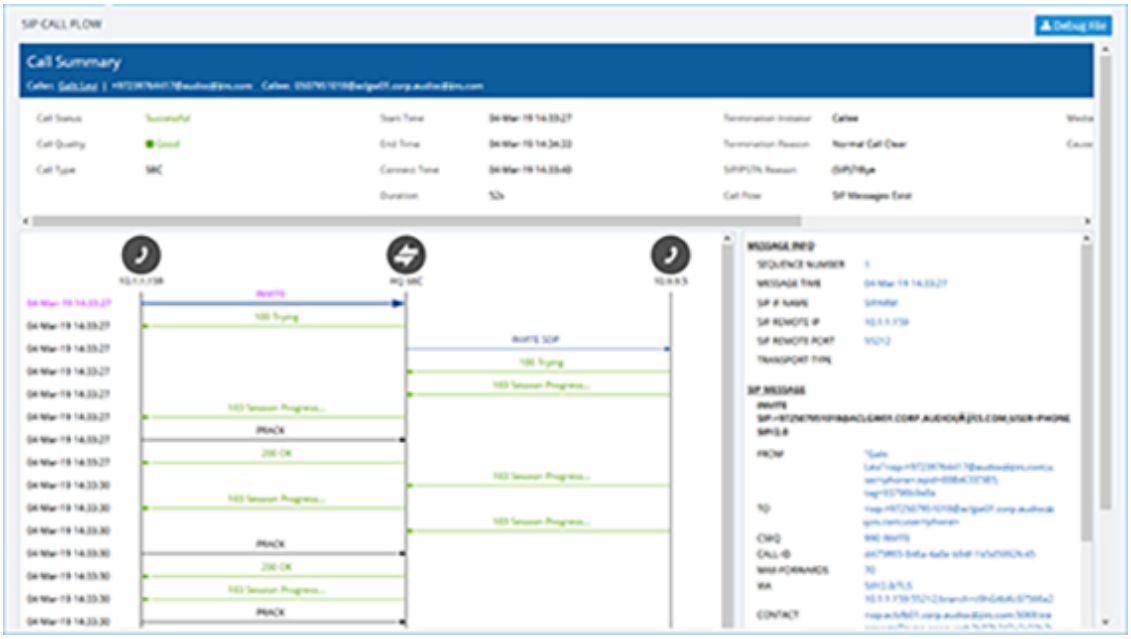
➢ **To configure debugging of Web services:**

1. Open the Web Service Settings page (**Setup** menu > **IP Network** tab > **Web Services** folder > **Web Service Settings**).
2. In the 'Debug Level' field (RestDebugMode), enter the debug level (or disable debugging by configuring it to 0):

3. Click **Apply**.

55 Enabling SIP Call Flow Diagrams in OVOC

You can configure the device to send SIP messages (in XML format) of SIP call dialogs to AudioCodes One Voice Operations Centers (OVOC) so that OVOC management users can view the call dialog as a call flow diagram. OVOC displays the call flow using vertical and horizontal lines where the vertical lines represent the SIP entities (including the device itself) involved in the dialog and where the horizontal lines represent the SIP requests and responses. An example of a SIP call flow diagram in OVOC is shown below.



SIP call flow diagrams may be useful for debugging and for better understanding of the SIP call. The call flow displays all the SIP messages related to the call session, including requests (e.g., INVITEs) and responses (e.g., 200 OK). For SBC calls, the call flow reflects messages as sent "over the wire" - incoming messages before manipulation and outgoing messages after manipulation.

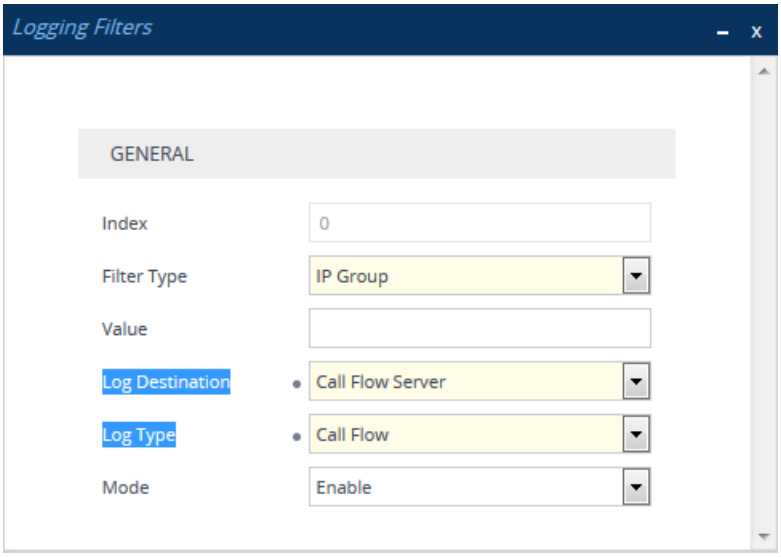
➢ To configure SIP call flow support:

1. Enable the OVOC call flow feature:
   a. Open the Logging Settings page (**Troubleshoot** menu > **Troubleshoot** tab > **Logging** folder > **Logging Settings**).
   b. From the 'Call Flow Report Mode' [CallFlowReportMode] drop-down list, select **Enable**.

2. To send call flow messages of specific calls only (e.g., for a specific IP Group), configure a Log Filter rule:
   a. Open the Logging Filters table (see Configuring Log Filter Rules on page 1059

b. Click **New** , and then configure the rule as desired, but with the following parameter settings:

- 'Log Destination': **Call Flow Server**
- 'Log Type': **Call Flow**



c. Click **Apply**.

- If you have not configured any filtering rule for SIP call flow in the Logging Filters table, the device sends call flow messages to OVOC for all calls.
- The device does not send OVOC SIP messages that fail authentication (SIP 4xx challenge).
- The feature does not support SIPRec messages and REGISTER messages.
- For HA systems, during a switchover the device stops sending the SIP call flow messages of current SIP dialogs and continues sending them after the switchover (even though OVOC does not display the continuation of the call after switchover).
- If the device experiences a CPU overload, it stops sending SIP call flow messages to OVOC until the CPU returns to normal levels.

56 Enabling Same Call Session ID over Multiple Devices

You can enable the use of a Global Session ID to identify call sessions traversing multiple devices. The Global Session ID is a randomly assigned ID number that identifies each call session. The ID is unique to the call session and remains the same throughout the session even if the call traverses multiple devices.

The Global Session ID appears in SIP messages using the AudioCodes proprietary SIP header, AC-Session-ID, as shown in the example below:

```
INVITE sip:2000@172.17.113.123;user=phone SIP/2.0
...
AC-Session-ID: 7f6941530b31d715
...
```

If the device receives an incoming SIP message containing the Global Session ID, it sends the same Global Session ID in the outgoing SIP message. If the incoming SIP message does not contain a Global Session ID or if a new session is initiated by the device, the device generates a new, unique Global Session ID and adds it to the outgoing SIP message.

To enable the Global Session ID, load an ini file to the device with the SendAcSessionIdHeader parameter configured to 1.

- The Global Session ID is not included in Syslog messages.
- By default, the device does not include the Global Session ID in CDRs. However, you can customize CDRs to include it. For more information, see Customizing CDRs for Gateway Calls and Customizing CDRs for SBC Calls and Test Calls on page 1036.
- If you disable this feature, the device sends outgoing SIP messages without a Global Session ID (even if a Global Session ID was received in the incoming SIP message).

57 Testing SIP Signaling Calls

A simulated endpoint can be configured on the device to test SIP signaling of calls between it and a remote destination. This feature is useful in that it can remotely verify SIP message flow without involving the remote end side in the debug process. The SIP test call simulates the SIP signaling process - call setup, SIP 1xx responses, through to completing the SIP transaction with a 200 OK.

The test call sends Syslog messages to a Syslog server, showing the SIP message flow, tone signals (e.g., DTMF), termination reasons, as well as voice quality statistics and thresholds (e.g., MOS).

Configuring Test Call Endpoints

The Test Call Rules table lets you test SIP signaling (setup and registration) and media (DTMF signals) of calls between a simulated phone on the device and a remote IP endpoint. These tests involve both incoming and outgoing calls, where the test endpoint can be configured as the caller or called party. The simulated phone and remote endpoints are defined as SIP URIs (user@host) and the remote endpoint can be defined as an IP Group or IP address.

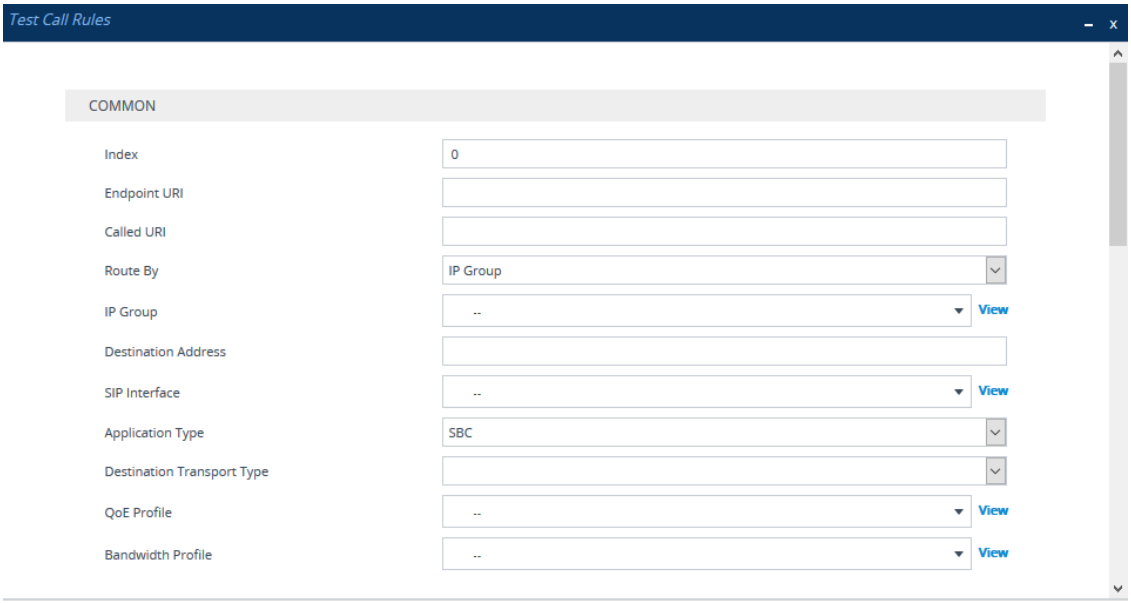
Test calls can be dialed automatically at a user-defined interval and/or manually when required. When a SIP test call is initiated, the device generates a SIP INVITE towards the remote endpoint (e.g., a SIP proxy server or softswitch). It simulates the SIP call setup process, managing SIP 1xx responses and completing the SIP transaction with a 200 OK.

- By default, you can configure up to five test call rules. However, you can increase this number by installing a License Key that licenses the required number. For more information, contact the sales representative of your purchased device.
- The device's Call Admission Control (CAC) feature (see Configuring Call Admission Control on page 696) does not apply to Test Calls.
- When the device operates in High-Availability (HA) mode, current Test Calls are disconnected during an HA switchover.

The following procedure describes how to configure test call rules through the Web interface. You can also configure it through ini file [Test_Call] or CLI (`configure troubleshoot > test-call test-call-table`).

➢ **To configure a test call rule:**

1. Open the Test Call Rules table (**Troubleshoot** menu > **Troubleshoot** tab > **Test Call** folder > **Test Call Rules**).
2. Click **New**; the following dialog box appears:



3. Configure a test call according to the parameters described in the table below.
4. Click **Apply**, and then save your settings to flash memory.

Table 57-1: Test Call Rules Table Parameter Descriptions

Parameter	Description
Common	
'Index'	Defines an index number for the new table row. **Note:** Each row must be configured with a unique index.
'Endpoint URI' `endpoint-uri` [Test_Call_EndpointURI]	Defines the endpoint's URI. This can be defined as a user or user@host. The device identifies this endpoint only by the URI's user part. The URI's host part is used in the SIP From header in REGISTER requests. The valid value is a string of up to 150 characters. By default, the parameter is not configured. **Note:** The parameter is mandatory.
'Called URI' `called-uri` [Test_Call_CalledURI]	Defines the destination (called) URI (user@host). The valid value is a string of up to 150 characters. By default, the parameter is not configured.
'Route By' `route-by` [Test_Call_RouteBy]	Defines the type of routing method. This applies to incoming and outgoing calls. ■ [1] **IP Group** = (Default) Calls are matched by (or routed to) an IP Group. To specify the IP Group, see the 'IP Group' parameter in the table.

Parameter	Description
	■ [2] **Dest Address** = Calls are matched by (or routed to) a destination IP address. To configure the address, see the 'Destination Address' parameter in the table. **Note:** ■ If configured to **Dest Address**: ✓ You must assign a SIP Interface (see the 'SIP Interface' below). ✓ The IP Profile of the default IP Group (ID 0) is used. You can use a different IP Profile, by specifying an IP Group in the 'IP Group' parameter (below). ■ For REGISTER messages, if configured to **IP Group**, only Server-type IP Groups can be used.
'IP Group' `ip-group-id` [Test_Call_IPGroupName]	Assigns an IP Group. This is the IP Group that the test call is sent to or received from. By default, no value is defined. To configure IP Groups, see Configuring IP Groups. **Note:** ■ The parameter is applicable if you configure the 'Route By' parameter to **IP Group**. ■ You can also use this parameter if you configure the 'Route By' parameter to **Dest Address**. This allows you to associate an IP Profile (that is assigned to the specified IP Group) with the Test Call. The Test Call is not routed to the IP Group, but uses only its IP Profile. ■ The IP Group is used for incoming and outgoing calls.
'Destination Address' `dst-address` [Test_Call_DestAddress]	Defines the destination host. The valid value is an IP address[:port] in dotted-decimal notation or a DNS name[:port]. **Note:** The parameter is applicable only if you configure the 'Route By' parameter to **Dest Address**.
'SIP Interface' `sip-interface-name`	Assigns a SIP Interface. This is the SIP Interface to which the test call is sent and received from.

Parameter	Description
[Test_Call_SIPInterfaceName]	By default, no value is defined. To configure SIP Interfaces, see Configuring SIP Interfaces. **Note:** The parameter is applicable only if the 'Route By' parameter is configured to **Dest Address**.
'Application Type' `application-type` [Test_Call_ApplicationType]	Defines the application type for the endpoint. This associates the IP Group and SRD to a specific SIP interface. ■ [2] **SBC** = SBC application **Note:** The parameter must always be configured to **SBC**.
'Destination Transport Type' `dst-transport` [Test_Call_DestTransportType]	Defines the transport type for outgoing calls. ■ [-1] = Not configured (default) ■ [0] **UDP** ■ [1] **TCP** ■ [2] **TLS** ■ [3] **SCTP** **Note:** The parameter is applicable only if you configure the 'Route By' parameter to **Dest Address**.
'QoE Profile' `qoe-profile` [Test_Call_QOEProfile]	Assigns a QoE Profile to the test call. By default, no value is defined. To configure QoE Profiles, see Configuring Quality of Experience Profiles.
'Bandwidth Profile' `bandwidth-profile` [Test_Call_BWProfile]	Assigns a Bandwidth Profile to the test call. By default, no value is defined. To configure Bandwidth Profiles, see Configuring Bandwidth Profiles.
Media	
'Offered Audio Coders Group' `offered-audio-coders-group-name` [Test_Call_ OfferedCodersGroupName]	Assigns a Coder Group, configured in the Coder Groups table, whose coders are added to the SDP Offer in the outgoing Test Call. If not configured, the device uses the Coder Group specified by the 'Extension Coders Group' parameter of

Parameter	Description
	the IP Profile associated with the rule's IP Group (see the 'IP Group' parameter above). To configure Coder Groups, see Configuring Coder Groups. **Note:** ■ The parameter's settings override the corresponding parameter of the IP Profile associated with the rule's IP Group. ■ If you don't configure this parameter nor the corresponding parameter of the associated IP Profile, the device uses Coder Group ID 0.
'Allowed Audio Coders Group' `allowed-audio-coders-group-name` [Test_Call_ AllowedAudioCodersGroupName]	Assigns an Allowed Audio Coders Group, configured in the Allowed Audio Coders Groups table, which defines only the coders that can be used for the test call. For incoming test calls, the device accepts the first offered coder that is supported and allowed. If not configured, the device uses the Allowed Audio Coders Group specified by the 'Allowed Audio Coders' parameter of the IP Profile associated with the rule's IP Group (see the 'IP Group' parameter above). To configure Allowed Audio Coders Groups, see Configuring Allowed Audio Coder Groups. **Note:** The parameter's settings override the corresponding parameter of the IP Profile associated with the rule's IP Group.
'Allowed Coders Mode' `allowed-coders-mode` [Test_Call_ AllowedCodersMode]	Defines the mode of the Allowed Coders feature for the Test Call. ■ [-1] **Not Configured** = (Default) The mode is according to the 'Allowed Coders Mode' parameter of the IP Profile associated with the rule's IP Group (see the 'IP Group' parameter above). ■ [0] **Restriction** = The device uses only allowed coders as configured in the 'Allowed Audio Coders Group' parameter (above) and removes all other coders from the SDP offer. If you have also configured additional coders in the 'Offered Audio Coders Group' parameter (above), then these coders are added to the SDP offer if they appear in the assigned Allowed Audio Coders Group.

Parameter	Description
	■ [1] **Preference** = The device re-arranges the priority (order) of the coders in the incoming SDP offer according to their order of appearance in the Allowed Audio Coders Group. The coders in the original SDP offer are listed after the allowed coders. ■ [2] **Restriction and Preference** = The device uses both the **Restriction** and **Preference** options. **Note:** Except for **Not Configured**, the parameter's settings override the corresponding parameter of the IP Profile associated with the rule's IP Group.
'Media Security Mode' `media-security-mode` [Test_Call_MediaSecurityMode]	Defines the handling of RTP and SRTP. ■ [-1] **Not Configured** = (Default) Handling is according to the 'SBC Media Security Mode' parameter of the IP Profile associated with the rule's IP Group (see the 'IP Group' parameter above). ■ [0] **As is** = No special handling for RTP\SRTP is done. ■ [1] **SRTP** = Only SRTP media lines are negotiated and RTP media lines are removed from the incoming SDP offer-answer. ■ [2] **RTP** = Only RTP media lines are negotiated and SRTP media lines are removed from the incoming SDP offer-answer. ■ [3] **Both** = Each SDP offer-answer is extended (if not already) to two media lines - one for RTP and one for SRTP. **Note:** ■ To enable SRTP, configure the [EnableMediaSecurity] parameter to [1]. ■ Except for **Not Configured**, the parameter's settings override the corresponding parameter of the IP Profile that is associated with the rule's IP Group.
'Play DTMF Method' `play-dtmf-method`	Defines the method used by the devicefor sending DTMF digits that are played to the called party when the call is answered.

Parameter	Description
[Test_Call_PlayDTMFMethod]	■ [-1] **Not Configured** = The mode is according to the 'Alternative DTMF Method' and 'RFC 2833 Mode' parameters of the IP Profile associated with the rule's IP Group (see the 'IP Group' parameter above). ■ [0] **RFC 2833** = (Default) The device sends the DTMF digits using the RFC 2833 method (out-of-band). ■ [1] **In Band** = The device sends the DTMF digits in-band (in the RTP stream). **Note:** ■ The parameter is applicable only if you configure the 'Play' parameter to **DTMF**. ■ Playing DTMF digits requires DSP resources when the DTMF method is **In Band**. ■ If the Test Call sends the SDP offer, the recommended DTMF configuration of the associated IP Profile is as follows: ✓ For RFC 2833: 'RFC 2833 Mode' = **Extend**; 'Alternative DTMF Method' = **As Is** ✓ For In Band: 'RFC 2833 Mode' = **Disallow**; 'Alternative DTMF Method' = **As Is** ■ If the Test Call receives the SDP offer, the recommended configuration is as follows (i.e., incoming SDP offer determines the method): 'RFC 2833 Mode' = **As Is**; 'Alternative DTMF Method' = **As Is** I
Authentication **Note:** These parameters are applicable only if the 'Call Party' parameter (below) is configured to **Caller.**	
'Auto Register' `auto-register` [Test_Call_AutoRegister]	Enables automatic registration of the endpoint. The endpoint can register to the device itself or to the 'Destination Address' or 'IP Group' parameter settings (see above). ■ [0] **Disable** (default) ■ [1] **Enable**

Parameter	Description
'Username' user-name [Test_Call_UserName]	Defines the authentication username. The valid value is a string of up to 60 characters. By default, no value is defined.
'Password' password [Test_Call_Password]	Defines the authentication password. By default, no password is defined. **Note:** The parameter cannot be configured with wide characters.
Test Setting	
'Call Party' call-party [Test_Call_CallParty]	Defines whether the test endpoint is the initiator (caller) or receiving side (called) of the test call. ■ [0] **Caller** (default) ■ [1] **Called**
'Maximum Channels for Session' max-channels [Test_Call_MaxChannels]	Defines the maximum number of concurrent channels for the test session. For example, if you have configured an endpoint "101" and you configure the parameter to "3", the device automatically creates three simulated endpoints - "101", "102" and "103" (i.e., consecutive endpoint URIs are assigned). The default is 1.
'Call Duration' call-duration [Test_Call_CallDuration]	Defines the call duration (in seconds). The valid value is -1 to 100000. The default is 20. A value of 0 means infinite. A value of -1 means that the parameter value is automatically calculated according to the values of the 'Calls per Second' and 'Maximum Channels for Session' parameters. **Note:** The parameter is applicable only if you configure 'Call Party' to **Caller**.
'Calls per Second' calls-per-second [Test_Call_CallsPerSecond]	Defines the number of calls per second. **Note:** The parameter is applicable only if you configure 'Call Party' to **Caller**.
'Test Mode' test-mode [Test_Call_TestMode]	Defines the test session mode. ■ [0] **Once** = (Default) The test runs until the lowest value between the following is reached: ✓ Maximum channels is reached for the test

Parameter	Description
	session, configured by 'Maximum Channels for Session'. ✓ Call duration ('Call Duration') multiplied by calls per second ('Calls per Second'). ✓ Test duration expires, configured by 'Test Duration'. ■ [1] **Continuous** = The test runs until the configured test duration is reached. If it reaches the maximum channels configured for the test session (in the 'Maximum Channels for Session'), it waits until the configured call duration of a currently established tested call expires before making the next test call. In this way, the test session stays within the configured maximum channels. **Note:** The parameter is applicable only if you configure 'Call Party' to **Caller**.
'Test Duration' `test-duration` [Test_Call_TestDuration]	Defines the test duration (in minutes). The valid value is 0 to 100000. The default is 0 (i.e., unlimited). **Note:** The parameter is applicable only if you configure 'Call Party' to **Caller**.
'Play' `play` [Test_Call_Play]	Enables the playing of a tone to the answered side of the call. ■ [0] **Disable** = No tone is played. ■ [1] **DTMF** = (Default) Plays (loop) a user-defined DTMF string, which is configured in Configuring DTMF Tones for Test Calls. ■ [2] **PRT** = Plays (loop) a pre-recorded tone (audio file) from the PRT file that is installed on the device. You can either specify the tone (by index) to play from the PRT file in the 'Play Tone Index' parameter (below), or implement a basic NetAnn feature whereby the tone from the PRT file (and other characteristics) are specified by NetAnn parameters in the Request-URI of the incoming SIP INVITE message. When using NetAnn, instead of connecting the call (i.e., 200 OK), the devicereplies with a SIP 183 containing SDP.

Parameter	Description
	The NetAnn parameters include the following: • **early=yes:** Indicates that NetAnn is used for playing the tone. • **play=<Prompt/Tone Index in PRT file>:** Defines the tone to play from the PRT file. • **repeat=<Times>:** Defines how many times the tone is played (loops) before the device disconnects the call. • **delay=<Delay Time>:** Defines the delay time (in msec) between each played (loop) tone. If the parameter is not present, the default is 2,000 ms (2 seconds). The following shows an example of a Request-URI with NetAnn parameters that instruct the device to play three times (loops) the tone that is defined at Index 15 in the PRT file: `INVITE sip:200@1.1.1.1;early=yes;play=15;repeat=3` **Note:** ■ You can configure the DTMF signaling type (RFC 2833 or in-band), using the 'Play DTMF Method' parameter (above). ■ Playing a tone from the PRT file requires DSP resources if the coder with which the tone was created is different to the coder used for the Test Call. ■ You can also use NetAnn parameters to play a specific recorded tone to the caller (source) when the destination fails for a regular IP-to-IP SBC call. To configure this: ✓ Configure a Message Manipulation rule that adds the NetAnn parameters, based on the tone that you want played, to the INVITE message's Request-URI. ✓ Configure a Number Manipulation rule that changes the destination number to the Test Call

Parameter	Description
	ID. ✓ Configure an IP Group to represent the device itself (which will be the Test Call module) and assign it the Message Manipulation rule and the Number Manipulation rule. ✓ Configure an alternative routing rule in the IP-to-IP Routing table that re-routes the call to the IP Group presenting the Test Call module. When the IP-to-IP call fails, the device uses the alternative routing rule to re-route the call to the Test Call module, which sends a SIP 183 response to the caller, playing the specified tone.
'Play Tone Index' `play-tone-index` [Test_Call_PlayToneIndex]	Defines the tone that you want played from the installed PRT file, to the called party when the call is answered. The valid value is the index number (1-80) of the tone in the PRT file. By default (-1), the device plays the tone defined at index 22 "acDialTone2". **Note:** ■ The parameter is applicable only if you configure the 'Play' parameter to **PRT**. ■ To play user-defined tones, you need to record your tones and then install them on the device using a loadable Prerecorded Tones (PRT) file, which is created using AudioCodes DConvert utility. When you create the PRT file, each recorded tone file must be added to the PRT file with the tone type "acUserDefineTone<Index>". For more information, see Prerecorded Tones File.
'Schedule Interval' `schedule-interval` [Test_Call_ScheduleInterval]	Defines the interval (in minutes) between automatic outgoing test calls. The valid value range is 0 to 100000. The default is 0 (i.e., scheduling is disabled). **Note:** The parameter is applicable only if you configure 'Call Party' to **Caller**.

Starting and Stopping Test Calls

The following procedure describes how to start, stop, and restart test calls.

➢ **To start, stop, and restart a test call:**

1. In the Test Call Rules table, select the required test call entry.
2. From the **Action** drop-down list, choose the required command:
   - **Dial:** Starts the test call (applicable only if the test call party is the caller).
   - **Drop Call:** Stops the test call.
   - **Restart:** Ends all established calls and then starts the test call session again.

Viewing Test Call Status

You can view the status of test call rules in the 'Test Status' field of the Test Call Rules table. The status can be one of the following:

Table 57-2: Test Call Status Description

Status	Description
"Idle"	Test call is not active.
"Scheduled"	Test call is planned to run (according to the 'Schedule Interval' parameter).
"Running"	Test call has been started (i.e., by clicking **Dial** from the 'Action' drop-down list).
"Receiving":	Test call has been automatically activated by calls received from the remote endpoint for the test call endpoint (when all these calls end, the status returns to "Idle").
"Terminating"	Test call is in the process of terminating currently established calls (when **Drop Call** is clicked from the 'Action' drop-down list to stop the test).
"Done"	Test call has successfully completed (or was prematurely stopped by clicking the **Drop Call** from the 'Action' drop-down list).

Viewing Test Call Statistics

You can view statistical information on the test call.

- On the receiving side, when the first call is accepted in "Idle" state, statistics are reset.
- The device also generates CDRs for test calls if you have enabled CDR generation (see Configuring CDR Reporting). To view CDRs of test calls, see Viewing CDR Test Calls .

➢ To view statistics of a test call:

1. In the Test Call Rules table, select the required test call row.
2. Scroll down the page to the area below the table. Statistics of the selected test call are displayed under the **Statistics** group, as shown in the example below:

STATISTICS	
Active Calls	0
Call Attempts	1
Total Established Calls	1
Total Failed Attempts	0
Remote Disconnections Count	1
Average CPS	1.00
Elapsed Time [HH:MM:SS]	00:00:20
Test Status	Done
Detailed Status	Done - Established Calls: 1, ASR: 100%
MOS Status	Local:N/A, Remote:N/A
Delay Status	Local:6 msec (Green), Remote:N/A
Jitter Status	Local:75 msec (Red), Remote:0 msec (Green)
Packet Loss Status	Local:0% (Green), Remote:0% (Green)
Bandwidth Status	Rx:0 KBytes/s (Green), Tx:0 KBytes/s (Green)

The statistics fields are described in the following table:

Table 57-3: Test Call Statistics Description

Statistics Field	Description
Active Calls	Number of currently established test calls.
Call Attempts	Number of calls that were attempted.
Total Established Calls	Total number of calls that were successfully established.
Total Failed Attempts	Total number of call attempts that failed.
Remote	Number of calls that were disconnected by the remote side.

Statistics Field	Description
Disconnections Count	
Average CPS	Average calls per second.
Elapsed Time	Duration of the test call since it was started (or restarted).
Test Status	Status (brief description) as displayed in the 'Test Status' field (see Viewing Test Call Status).
Detailed Status	Displays a detailed description of the test call status: ■ "Idle": Test call is currently not active. ■ "Scheduled - Established Calls: <number of established calls>, ASR: <ASR>%": Test call is planned to run (according to 'Schedule Interval' parameter settings) and also shows the following summary of completed test calls: ✓ Total number of test calls that were established. ✓ Number of successfully answered calls out of the total number of calls attempted (ASR). ■ "Running (Calls: <number of active calls>, ASR: <ASR>%)": Test call has been started (i.e., the **Dial** command was clicked) and shows the following: ✓ Number of currently active test calls. ✓ Number of successfully answered calls out of the total number of calls attempted (Answer Success Ratio or ASR). ■ "Receiving (<number of active calls>)": Test call has been automatically activated by calls received for this configured test call endpoint from the configured remote endpoint. When all these calls terminate, the status returns to "Idle". ■ "Terminating (<number of active calls>)": The **Drop Call** command has been clicked to stop the test call and the test call is in the process of terminating the currently active test calls. ■ "Done - Established Calls: <number of established calls>, ASR: <ASR>%": Test call has been successfully completed (or was prematurely stopped by clicking the **Drop Call** command) and shows the following: ✓ Total number of test calls that were established. ✓ Number of successfully answered calls out of the total

Statistics Field	Description
	number of calls attempted (ASR).
MOS Status	MOS count and color threshold status of local and remote sides according to the assigned QoE Profile.
Delay Status	Packet delay count and color-threshold status of local and remote sides according to the assigned QoE Profile.
Jitter Status	Jitter count and color-threshold status of local and remote sides according to the assigned QoE Profile.
Packet Loss Status	Packet loss count and color-threshold status of local and remote sides according to the assigned QoE Profile.
Bandwidth Status	Tx/Rx bandwidth and color-threshold status according to the assigned Bandwidth Profile.

Configuring DTMF Tones for Test Calls

By default, the device plays the DTMF signal tone "3212333" to remote tested endpoints for answered calls (incoming and outgoing). For basic test calls (as described in Configuring Basic Test Calls), the device can play only the configured DTMF tones (or none, if not configured). For test call endpoints that are configured in the Test Call Rules table, you can configure the device to play either DTMF tones or a tone from an installed PRT file (Test Call Tone). For more information, see Configuring Test Call Endpoints.

- You can configure the DTMF signaling type (e.g., out-of-band or in-band) using the 'DTMF Transport Type' [DTMFTransportType] parameter.
- To generate DTMF tones, the device's DSP resources are required.

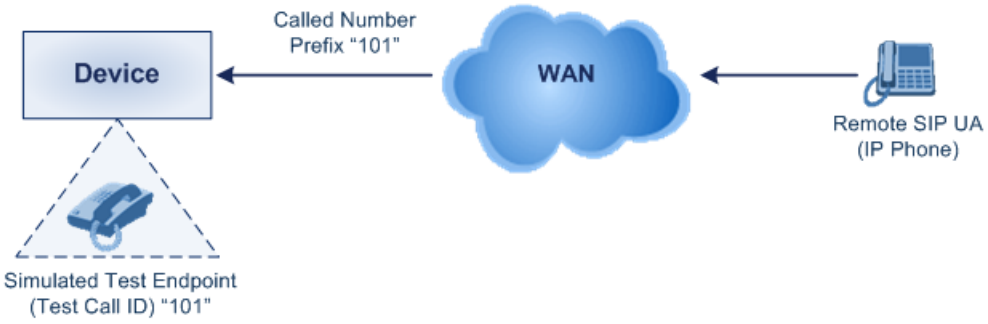
➢ **To configure played DTMF signal to answered test call:**

1. Open the Test Call Settings page (**Troubleshoot** menu > **Troubleshoot** tab > **Test Call** folder > **Test Call Settings**).
2. In the 'Test Call DTMF String' field, enter the DTMF string (up to 15 digits):

3. Click **Apply**.

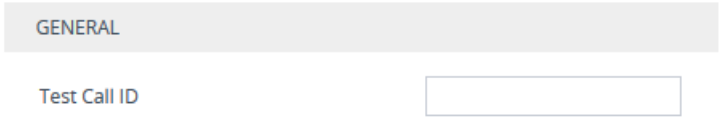
Configuring Basic Test Calls

The Basic Test Call feature tests **incoming** calls from remote SIP (IP) endpoints to a single simulated test endpoint on the device. The only required configuration is to assign a prefix number (*test call ID*) to the simulated endpoint. Incoming calls with this called (destination) prefix number are identified by the device as test calls and sent to the simulated endpoint. The figure below displays a basic test call example:



➢ **To configure basic call testing:**

1. Open the Test Call Settings page (**Troubleshoot** menu > **Troubleshoot** tab > **Test Call** folder > **Test Call Settings**).
2. In the 'Test Call ID' field, enter a prefix for the simulated endpoint:



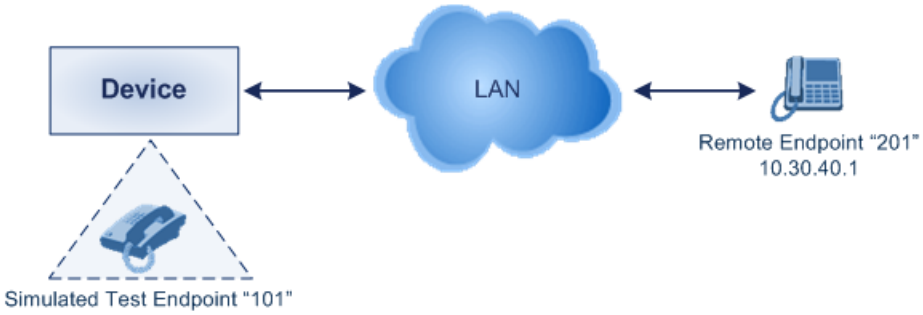
   For a full description of the parameter, see SIP Test Call Parameters.
3. Click **Apply**.

- The device can play DTMF tones to the remote endpoint. For more information, see Configuring DTMF Tones for Test Calls.
- The Basic Test Call feature uses the default IP Group (ID #0) and its associated IP Profile (if exists).
- Test calls are done on all SIP Interfaces.

Test Call Configuration Examples

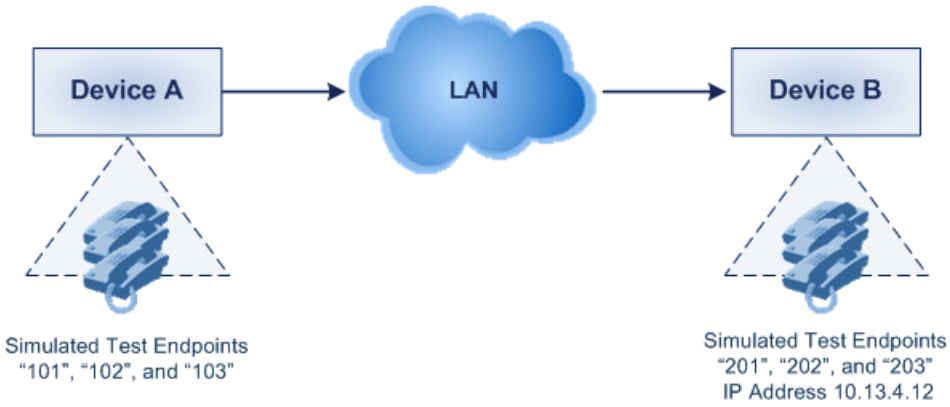
Below are a few examples of test call configurations.

- **Single Test Call Scenario:** This example describes the configuration of a simple test call scenario that includes a single test call between a simulated test endpoint on the device and a remote endpoint.



- Test Call Rules table configuration:
  - Endpoint URI: "101"
  - Called URI: "201"
  - Route By: **Dest Address**
  - Destination Address: "10.30.40.01"
  - SIP Interface: SIPInterface_0
  - Call Party: **Caller**
  - Test Mode: **Once**

- **Batch Test Call Scenario:** This example describes the configuration of a batch test call setup for scheduled and continuous call testing of multiple endpoints. The test call is done between two AudioCodes devices - Device A and Device B - with simulated test endpoints. This eliminates the need for phone users, who would otherwise need to answer and end calls many times for batch testing. The calls are initiated from Device A, where Device B serves as the remote answering endpoint.

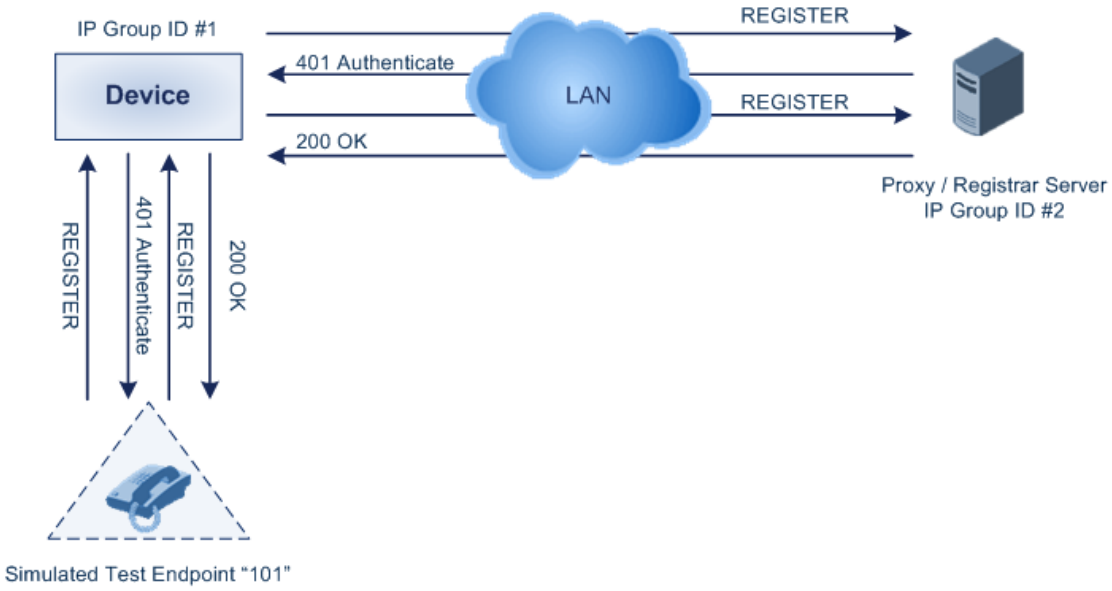


  - Test Call Rules table configuration at Device A:
    - Endpoint URI: "101"
    - Called URI: "201"
    - Route By: **Dest Address**
    - Destination Address: "10.13.4.12"
    - SIP Interface: SIPInterface_0

- Call Party: **Caller**
- Maximum Channels for Session: "3" (configures three endpoints - "101", "102" and "103)
- Call Duration: "5" (seconds)
- Calls per Sec: "1"
- Test Mode: **Continuous**
- Test Duration: "3" (minutes)
- Schedule Interval: "180" (minutes)

- Test Call Rules table configuration at Device B:
  - Endpoint URI: "201"
  - Maximum Channels for Session: "3" (configures three endpoints - "201", "202" and "203)

Registration Test Call Scenario: This example describes the configuration for testing the registration and authentication (i.e., username and pas,sword) process of a simulated test endpoint on the device with an external proxy/registrar server. This is useful, for example, for verifying that endpoints located in the LAN can register with an external proxy and subsequently, communicate with one another.



This example assumes that you have configured your device for communication between LAN phone users such as IP Groups to represent the device (10.13.4.12) and the proxy server, and IP-to-IP routing rules to route calls between these IP Groups.

- Test Call Rules table configuration:
  - Endpoint URI: "101"
  - Called URI: "itsp"

- Route By: **Dest Address**
- Destination Address: "10.13.4.12" (this is the IP address of the device itself)
- SIP Interface: SIPInterface_0
- Auto Register: **Enable**
- User Name: "testuser"
- Password: "12345"
- Call Party: **Caller**

58 Pinging a Remote Host or IP Address

You can verify the network connectivity with a remote host or IP address by pinging the network entity.

- IPv4: The ping to an IPv4 address can be done from any of the device's VoIP interfaces that is configured with an IPv4 address. The ping is done using the following CLI command:

```
# ping <IPv4 ip address or hostname> source [voip] interface
```

For a complete description of the ping command, refer to the *CLI Reference Guide*.

Part X

Appendix

59 Patterns for Denoting Phone Numbers and SIP URIs

The table below lists the supported patterns (notations) that you can use in various configuration tables for matching rules, based on source and/or destination phone numbers and SIP URIs (user@host parts).

When configuring phone numbers in the Web interface, enter them only as digits without any other characters. For example, if you wish to enter the phone number 555-1212, type it as 5551212 without the hyphen (-). If the hyphen is entered, the entry is invalid.

Table 59-1: Supported Patterns for Phone Numbers and SIP URIs

Pattern	Description
x (letter "x")	Wildcard that denotes any single digit or character.
# (pound symbol)	■ When located at the end of a pattern, it denotes the end of a number. For example, **54324#** denotes a 5-digit number that starts with the digits 54324. ■ When located anywhere in the pattern except at the end, it is part of the number (pound key). For example, **3#45** represents the prefix number 3#45. ■ To denote the # key when it appears at the end of the number, enclose it in square brackets. For example, 134[#] denotes any number that starts with 134#.
* (asterisk symbol)	■ When used on its own, it denotes any number or string. ■ When used as part of a number, it denotes the asterisk (*) key. For example, *345 denotes a number that starts with *345.
$ (dollar sign)	For incoming IP calls: Denotes a Request-URI that does not have a user part. This pattern is used for the following matching criteria: ■ Source and Destination Phone ■ Source and Destination Username ■ Source and Destination Calling Name
Range of Digits	**Note:** ■ To denote a prefix that is a range, enclose it in square brackets, for

Pattern	Description
	example, **[4-8]** or **23xx[456]**. ■ To denote a prefix that is not a range, do not enclose in brackets, for example, **12345#**. ■ To denote a suffix, enclose it in parenthesis, for example, **(4)** and **(4-8)**. ■ To denote a suffix that includes multiple ranges, enclose the range in square brackets, for example, **(23xx[4,5,6])**. Example of using both a prefix and a suffix in a pattern: Assume you want to match a rule whose destination phone prefix is 4 through 8, and suffix is 234, 235, or 236. The pattern for this would be: **[4-8](23[4,5,6])**.
[n-m] or **(n-m)**	Denotes a range of numbers. Examples: ■ To denote prefix numbers from 5551200 to 5551300: ✓ **[5551200-5551300]#** ■ To denote prefix numbers from 123100 to 123200: ✓ **123[100-200]#** ■ To denote prefix and suffix numbers together: ✓ 03(100): for any number that starts with 03 and ends with 100. ✓ [100-199](100,101,105): for a number that starts with 100 to 199 and ends with 100, 101 or 105. ✓ 03(abc): for any number that starts with 03 and ends with abc. ✓ 03(5xx): for any number that starts with 03 and ends with 5xx. ✓ 03(400,401,405): for any number that starts with 03 and ends with 400 or 401 or 405. **Note:** ■ The value *n* must be less than the value *m*. ■ Only numerical ranges are supported (not letters). ■ For suffix ranges, the starting (*n*) and ending (*m*) numbers in the range must include the same number of digits. For example, (23-34) is correct, but (3-12) is not.
[n,m] or **(n,m)**	Denotes multiple numbers. The value can include digits or characters. Examples:

Pattern	Description
	■ To denote a one-digit number starting (prefix) with 2, 3, 4, 5, or 6: **[2,3,4,5,6]** ■ To denote a one-digit number ending (suffix) with 7, 8, or 9: **(7,8,9)** ■ Prefix with suffix: **[2,3,4,5,6](7,8,9)** - prefix is denoted in square brackets; suffix in parenthesis For **prefix only**, the patterns *d[n,m]e* and *d[n-m]e* can also be used: ■ To denote a five-digit number that starts with 11, 22, or 33: **[11,22,33]xxx#** ■ To denote a six-digit number that starts with 111 or 222: **[111,222]xxx#**
[n1-m1,n2-m2,a,b,c,n3-m3] or **(n1-m1,n2-m2,a,b,c,n3-m3)**	Denotes a mixed pattern of single numbers and multiple number ranges. For example, to denote numbers 123 through 130, 455, 766, and 780 through 790: ■ Prefix: **[123-130,455,766,780-790]** ■ Suffix: **(123-130,455,766,780-790)** **Note:** The ranges and the single numbers in the mixed pattern must have the same number of digits. For example, each number range and single number in the examples above consists of three digits (e.g., 780).
Special ASCII Characters	The device does not support the use of ASCII characters in manipulation rules and therefore, for LDAP-based queries, the device can use the hexadecimal (HEX) format of the ASCII characters for phone numbers instead. The HEX value must be preceded by a backslash "\". For example, you can configure a manipulation rule that changes the received number **+49 (7303) 165-xxxxx** to **+49 \287303\29 165-xxxxx**, where *\28* is the ASCII HEX value for "(" and *\29* is the ASCII HEX value for ")". The manipulation rule in this example would denote the parenthesis in the destination number prefix using "x" wildcards (e.g., xx165xxxxx#); the prefix to add to the number would include the HEX values (e.g., +49 \287303\29 165-). Below is a list of common ASCII characters and their corresponding HEX values: ■ ***:** \2a ■ **(:** \28 ■ **):** \29 ■ **\:** \5c

Pattern	Description
	■ **/:** \2f

60 Configuration Parameters Reference

The device's VoIP functionality configuration parameters, default values, and their descriptions are documented in this section.

Parameters and values enclosed in square brackets [...] represent ini file parameters and their enumeration values.

Management Parameters

This section describes the device's management-related parameters.

General Parameters

The general management parameters are described in the table below.

Table 60-1: General Management Parameters

Parameter	Description
'Host Name' `configure system > hostname` [Hostname]	Defines a hostname for the device. This can be used, for example, to access the Web interface instead of using the IP address. The valid value is a string of up to 18 characters. By default, no value is defined. For more information, see Configuring a Hostname for the Device on page 109.
[WebLoginBlockAutoComplete]	Disables autocompletion when entering the management login username in the 'Username' field of the device's Web interface. Disabling autocompletion may be useful for security purposes by hiding previously entered usernames and thereby, preventing unauthorized access to the device's management interface. ■ [0] **Disable** = (Default) Autocompletion is enabled and the 'Username' field automatically offers previously logged in usernames. ■ [1] **Enable** = Autocompletion is disabled.
`configure system > web > enforce-password-complexity` [EnforcePasswordComplexity]	Enables the enforcement of the management user's login password complexity requirements to ensure strong passwords. ■ [0] **Disable** (default)

Parameter	Description
	■ [1] **Enable** For more information on password complexity requirements, see the 'Password' parameter in Configuring Management User Accounts.
`configure system > web > min-web-password-len` [MinWebPasswordLen]	Defines the minimum length (number of characters) of the management user's login password when password complexity is enabled (using the [EnforcePasswordComplexity] parameter). The valid value is a string of 8 to 20 characters. The default is 8. For more information on password complexity requirements, see the 'Password' parameter in Configuring Management User Accounts.
'Lock' `admin state lock` [AdminState]	Locks the device, whereby existing calls are terminated (optionally, after a graceful period) and new calls are rejected. ■ [0] **Lock** ■ [2] **Unlock** (default) For more information, see Locking and Unlocking the Device on page 852.
'Graceful Option' `admin state lock graceful` [AdminStateLockControl]	Defines a graceful period (in seconds) before the device locks. During this period, the device does not accept any new calls, allowing only existing calls to continue until the timeout expires. If all existing calls end before the timeout expires, the device locks. If there are still existing calls when the timeout expires, the device terminates them and then locks. The valid value is 0 to 32,768 seconds. The default is 0, meaning no graceful lock (i.e., immediate lock). A value of -1 means that the device locks only after all the existing calls end (on their own accord). For more information, see Locking and Unlocking the Device on page 852.
'Disconnect Client Connections' `admin state lock no-graceful disconnect-`	Enables the device to close existing TLS/TCP client connections and reject incoming TLS/TCP client connections when the device is in locked state.

Parameter	Description
`client-connections` [AdminStateRestrictConnections]	■ [0] **Disable** (default) ■ [1] **Enable** For more information, see Locking and Unlocking the Device on page 852.
'Floating License' `configure system > floating-license > floating-license` [EnableFloatingLicense]	Enables the device to operate with the Floating License. ■ [0] **Disable** (default) ■ [1] **Enable** For more information, see Floating License Model on page 884. **Note:** For the parameter to take effect, a device reset is required.
'Allocation Profile' `configure system > floating-license > allocation-profile` [AllocationProfile]	Defines an SBC capacity profile (Allocation Profile) for the Floating License feature. ■ [0] **SIP Trunking** = (Default) Profile suited for SIP Trunking applications. ■ [1] **Registered Users** = Profile suited for applications requiring registered users. ■ [2] **Custom** = Customized Allocation Profile. **Note:** For the parameter to take effect, a device reset is required.
'Allocation - Far End Users' `configure system > floating-license > allocation-registered-users` [AllocationRegisteredUsers]	Defines registered users capacity for a customized Allocation Profile for the Floating License feature. **Note:** ■ For the parameter to take effect, a device reset is required. ■ The parameter is applicable only when the 'Allocation Profile' is configured to **Custom**.
'Allocation – SBC Media Sessions' `configure system > floating-license > allocation-media-sessions` [AllocationMediaSessions]	Defines SBC media session capacity for a customized Allocation Profile for the Floating License feature. **Note:** ■ For the parameter to take effect, a device reset is required.

Parameter	Description
	■ The parameter is applicable only when the 'Allocation Profile' is configured to **Custom**.
'Allocation – SBC Signaling Sessions' `configure system > floating-license > allocation-signaling-sessions` [AllocationSignalingSessions]	Defines SBC signaling session capacity for a customized Allocation Profile for the Floating License feature. **Note:** ■ For the parameter to take effect, a device reset is required. ■ The parameter is applicable only when the 'Allocation Profile' is configured to **Custom**.
'Limit - Far End Users' `configure system > floating-license > limit-registered-users` [LimitRegisteredUsers]	Defines a limit of the registered user capacity for a customized Allocation Profile for the Floating License feature. **Note:** The parameter is applicable only when the 'Allocation Profile' is configured to **Custom**.
'Limit – SBC Media Sessions' `configure system > floating-license > limit-media-sessions` [LimitMediaSessions]	Defines a limit of the SBC media session capacity for a customized Allocation Profile for the Floating License feature. **Note:** The parameter is applicable only when the 'Allocation Profile' is configured to **Custom**.
'Limit – SBC Signaling Sessions' `configure system > floating-license > limit-signaling-sessions` [LimitSignalingSessions]	Defines a limit of the SBC SIP signaling session capacity for a customized Allocation Profile for the Floating License feature. **Note:** The parameter is applicable only when the 'Allocation Profile' is configured to **Custom**.
'Limit – Transcoding Sessions' `configure system > floating-license > limit-transcoding-sessions` [LimitTranscodingSessions]	Defines a limit of the transcoding session capacity for a customized Allocation Profile for the Floating License feature. **Note:** The parameter is applicable only when the 'Allocation Profile' is configured to **Custom**.
[CustomerSN]	Defines a serial number (S/N) for the device. **Note:** The device's original S/N is automatically added at the end of the configured S/N. For example, if the original S/N is 8906721 and the configured S/N is "abc123", the resultant S/N is

Parameter	Description
	"abc1238906721".

Web Parameters

The Web parameters are described in the table below.

Table 60-2: Web Parameters

Parameter	Description
'Enable web access from all interfaces' `web-access-from-all-interfaces` [EnableWebAccessFromAllInterfaces]	Enables Web access from any of the device's IP network interfaces. The feature applies to HTTP and HTTPS protocols. ■ [0] = (Default) Disable – Web access is only through the OAMP interface. ■ [1] = Enable - Web access is through any network interface. **Note:** ■ For the parameter to take effect, a device reset is required. ■ Instead of using this parameter, you can use the Additional Management Interfaces table to assign specific IP network interfaces for management interfaces (as well as assign them TLS Contexts). For more information, see Configuring Additional Management Interfaces.
'Password Change Interval' `web-password-change-interval` [WebPassChangeInterval]	Defines the minimum duration (in minutes) between login password changes. For example, if you configure the parameter to 60, you can only change the password if at least 60 minutes has elapsed since the password was last changed. The valid value is 0 to 100,000. The default is 0, meaning that the password can be changed at any time.
'User Inactivity Timer' [UserInactivityTimer]	Defines the duration (in days) for which a user has not logged in to the Web interface, after which the status of the user becomes inactive and can no longer access the Web interface. These users can only log in to the Web interface if their status is changed (to New or Valid) by a Security Administrator or Master user.

Parameter	Description
	The valid value is 0 to 10000, where 0 means inactive. The default is 90. **Note:** The parameter is applicable only when using the Local Users table.
'Session Timeout' [WebSessionTimeout]	Defines the duration (in minutes) of inactivity of a logged-in user in the Web interface, after which the user is automatically logged off the Web session. In other words, the session expires when the user has not performed any operations (activities) in the Web interface for the configured duration. The valid value is 0, or 2 to 100000, where 0 means no timeout. The default is 15. **Note:** You can also configure the feature per user in the Local Users table (see Configuring Management User Accounts), which overrides this global setting.
'Deny Access On Fail Count' [DenyAccessOnFailCount]	Defines the maximum number of failed login attempts, after which the requesting IP address (management station) for all users is blocked. The valid value range is 0 to 10. The value 0 means that the feature is disabled and no blocking is done. The default is 3.
'Deny Authentication Timer' [DenyAuthenticationTimer]	Defines the duration (in seconds) for which login to the Web interface is denied from a specific IP address (management station) for all users when the number of failed login attempts exceeds the maximum. This maximum is configured by the [DenyAccessOnFailCount] parameter. Only after this time expires can users attempt to log in from this same IP address. The valid value is 0 to 100000, where 0 means that login is not denied regardless of the number of failed login attempts. The default is 60. **Note:** ■ If the [BlockingDurationFactor] parameter is configured to a value greater than 1, the duration that the IP address is blocked is according to the [BlockingDurationFactor] parameter. ■ To configure the duration for which the IP address is blocked, use the [DenyAuthenticationTimer]

Parameter	Description
	parameter. ■ Up to 1,000 IP addresses (management stations) can be blocked concurrently.
'Blocking Duration Factor' `blocking-duration-factor` [BlockingDurationFactor]	Defines the number to multiple the previous blocking time for blocking the IP address (management station) or user upon the next failed login scenario. The valid value is 1 to 100. The default is 1, which means that the blocking time remains the same (not increased). For example, assume the following configuration: ■ The 'Deny Access On Fail Count' parameter is configured to 3 (failed login attempts). ■ The [WebUsers_BlockTime] parameter (Local Users table) is configured to 10 (seconds) for user blocking (or [DenyAuthenticationTimer] parameter is configured to 10 for IP address blocking). ■ The [BlockingDurationFactor] parameter is configured to 2. After three failed login attempts, the device blocks the user for 10 seconds. If the user tries again to login but fails after three attempts, the device blocks the user for 20 seconds (i.e., 10 x 2). If the user tries again to login but fails after three attempts, the device blocks the user for 40 seconds (i.e., 20 x 2), and so on.
'Valid time of Deny Access counting' `deny-access-counting-valid-time` [DenyAccessCountingValidTime]	Defines the maximum time interval (in seconds) between failed login attempts to be included in the count of failed login attempts for denying access to the user. The valid value is 30 to 10,000,000. The default is 60. For example, assume the following: ■ The [DenyAccessOnFailCount] parameter is configured to 3 (failed login attempts). ■ The [DenyAccessCountingValidTime] parameter is configured to 30 (seconds). If the user makes a failed login attempt, and then makes another failed login attempt 32 seconds later, and another failed login attempt 10 seconds later, the

Parameter	Description
	user is not blocked by the device. This is because the interval between the first and second attempt was greater than the 30 seconds configured for the [DenyAccessCountingValidTime] parameter. However, if the interval between all three failed login attempts is less than 30 seconds, the device blocks the user.
'Display Last Login Information' [DisplayLoginInformation]	Enables the display of the user's login information upon each successful login attempt. ■ [0] **Disable** (default) ■ [1] **Enable**
'Enable Management Two Factor Authentication' [EnableMgmtTwoFactorAuthentication]	Enables Web login authentication using a third-party, smart card (two-factor authentication). ■ [0] Disable (default) ■ [1] Enable For more information, see Web Login Authentication using Smart Cards on page 67.
[CSRFProtection]	Enables cross-site request forgery (CSRF) protection of the device's embedded Web server. ■ [0] = Disable ■ [1] = (Default) Enable For more information, see Enabling CSRF Protection on page 66.
`http-port` [HTTPport]	Defines the LAN HTTP port for Web management. To enable Web management from the LAN, configure the desired port. The default is 80. **Note:** For the parameter to take effect, a device reset is required.
[DisableWebConfig]	Defines if the entire Web interface is read-only. ■ [0] = (Default) Enables modifications of parameters. ■ [1] = Web interface is read-only. When in read-only mode, parameters can't be modified and the following pages can't be accessed: Web User

Parameter	Description
	Accounts, TLS Contexts, Time and Date, Maintenance Actions, Load Auxiliary Files, Software Upgrade Wizard, and Configuration File. **Note:** ■ For the parameter to take effect, a device reset is required.
[ResetWebPassword]	Enables the device to restore the default management user accounts. ■ [0] = (Default) Disabled - currently configured user accounts (usernames and passwords) are retained. ■ [1] = Enabled - default user accounts (listed below) are restored and all other configured users (in the Local Users table) are deleted: ✓ Security Administrator user (username **Admin** and password **Admin**) ✓ Monitor user (username **User** and password **User**) **Note:** ■ You can also restore the default management user accounts (and delete all other configured users) through SNMP, by setting acSysGenericINILine to "ResetWebPassword = 1". ■ You can change username and password through SNMP: ✓ To change the current username, use the following syntax (but without angled brackets) in the acSysWEBAccessEntry table: acSysWEBAccessUserName:<current username>/<password>/<new username> ✓ To change the current password, use the following syntax (but without angled brackets) in the acSysWEBAccessEntry table:

Parameter	Description
	acSysWEBAccessUserCode:<username>/<current password>/<new password>
[WelcomeMessage] `configure system > welcome-msg`	Defines a welcome message displayed on the Web interface's Web Login page. The format of the ini file table parameter is: [WelcomeMessage] FORMAT WelcomeMessage_Index = WelcomeMessage_Text [\WelcomeMessage] For Example: FORMAT WelcomeMessage_Index = WelcomeMessage_Text WelcomeMessage 1 = "**********" ; WelcomeMessage 2 = "********* This is a Welcome message ***" ; WelcomeMessage 3 = "**********" ; For more information, see Creating a Login Welcome Message. **Note:** ■ Each index row represents a line of text. Up to 20 lines (or rows) of text can be defined. ■ The configured text message must be enclosed in double quotation marks (i.e., "..."). ■ If the parameter is not configured, no Welcome message is displayed.
[UseProductName]	Enables the option to customize the name of the device (product) that appears in the management interfaces. ■ [0] = Disabled (default). ■ [1] = Enables the display of a user-defined name, which is configured by the [UserProductName] parameter. For more information, see Customizing the Product Name.

Parameter	Description
[UserProductName]	Defines a name for the device instead of the default name. The value can be a string of up to 29 characters. For more information, see Customizing the Product Name. **Note:** To enable customization of the device name, see the [UseProductName] parameter.
[UseWebLogo]	Enables the Web interface to display user-defined text instead of an image (logo). ■ [0] = (Default) The Web interface displays a logo image, configured by the [LogoFileName] parameter. ■ [1] = The Web interface displays text (instead of an image), configured by the [WebLogoText] parameter (see note). For more information, see Replacing the Corporate Logo. **Note:** If you want to display text instead of an image, configure [UseWebLogo] to 1 and make sure that [LogoFileName] is not configured to any value. If [LogoFileName] is configured, it overrides [UseWebLogo] and an image will always be displayed.
[WebLogoText]	Defines the text that is displayed instead of the logo in the Web interface. The valid value is a string of up to 15 characters. For more information, see Replacing the Corporate Logo with Text. **Note:** The parameter is applicable only when the [UseWebLogo] parameter is configured to [1].
[LogoFileName]	Defines the name of the image file that you want loaded to the device. This image is displayed as the logo in the Web interface (instead of the AudioCodes logo). The file name can be up to 47 characters. For more information, see Replacing the Corporate Logo with an Image. **Notes:** ■ The image file type can be one of the following: GIF,

Parameter	Description
	PNG, JPG, or JPEG. ■ The size of the image file can be up to 64 Kbytes.

Telnet and CLI Parameters

The Telnet parameters are described in the table below.

Table 60-3: Telnet Parameters

Parameter	Description
'Embedded Telnet Server' `configure system > cli-settings > telnet` [TelnetServerEnable]	Enables the device's embedded Telnet server. ■ [0] **Disable** ■ [1] **Enable Unsecured** (default) ■ [2] **Enable Secured** **Note:** Only management users with Security Administrator or Master user levels can access the CLI's Privileged command mode.
'Telnet Server TCP Port' `configure system > cli-settings > telnet-port` [TelnetServerPort]	Defines the port number for the embedded Telnet server. The valid range is all valid port numbers. The default port is 23.
'Telnet Server Idle Timeout' `configure system > cli-settings > idle-timeout` [TelnetServerIdleDisconnect]	Defines the duration of an idle CLI (Telnet or SSH) session after which the session is automatically disconnected. The valid range is any value. The default is 5. When configured to 0, idle sessions are not disconnected. **Note:** If you change the parameter's value when there are current Telnet/SSH sessions, the parameter's previous setting is still applied to these current sessions and the parameter's new setting is applied only to new sessions.
'Maximum Telnet Sessions' `configure system > cli-settings > telnet-max-sessions` [TelnetMaxSessions]	Defines the maximum number of permitted, concurrent Telnet or SSH sessions. The valid range is 1 to 5 sessions. The default is 2. **Note:** Before changing the value, make sure that not more than this number of sessions are currently active; otherwise, the new setting will not take

Parameter	Description
	effect.
[CLIEnableModePassword]	Defines the password to access the Enable configuration mode in the CLI. The valid value is a string of up to 50 characters. The default is "Admin". **Note:** ■ The password is case-sensitive. ■ The parameter cannot be configured with wide characters.
'Default Terminal Window Height' `configure system > cli-settings > default-window-height` [DefaultTerminalWindowHeight]	Defines the number (height) of output lines displayed in the CLI terminal window. This applies to all new CLI sessions and is preserved after device resets. The valid value range is -1 (default) and 0-65535: ■ A value of -1 means that the parameter is disabled and the settings of the CLI command **window-height** is used. ■ A value of 0 means that all the CLI output is displayed in the window. ■ A value of 1 or greater displays that many output lines in the window and if there is more output, the "—MORE—" prompt is displayed. For example, if you configure the parameter to 4, up to four output lines are displayed in the window and if there is more output, the "—MORE—" prompt is displayed (at which you can press the spacebar to display the next four output lines). **Note:** You can override this parameter for a specific CLI session and configure a different number of output lines, by using the **window-height** CLI command in the currently active CLI session.
`configure system > mgmt-auth > obscure-password-mode` [CliObscuredPassword]	Enables the device to enforce obscured (i.e., encrypted) passwords whenever you create a new management user or modify the password of an existing user, through CLI (`configure system > user`). ■ [0] = (Default) Disabled - passwords are

Parameter	Description
	configured in plain text. ■ [1] = Enabled - passwords must be configured in encrypted format. To obtain an encrypted (obscured) password: a. Enable the parameter. b. Configure the user's password in the Web interface's Local Users table (see Configuring Management User Accounts on page 48). c. Save the device's CLI Script file to your local PC (see Saving and Loading CLI Script Files on page 896). d. Open the file, and then copy the encrypted password to the CLI where you are configuring the user.

ini File Parameters

The parameters relating to ini-file management are described in the table below.

Table 60-4: ini File Parameters

Parameter	Description
[INIPasswordsDisplayType]	Defines how passwords are displayed in the ini file. ■ [0] = (default) Disable. Passwords are obscured ("encoded"). The passwords are displayed in the following syntax: 1<obscured password> (e.g., 1S3p+fno=). ■ [1] = Enable. All passwords are hidden and replaced by an asterisk (*).

SNMP Parameters

The SNMP parameters are described in the table below.

Table 60-5: SNMP Parameters

Parameter	Description
'Disable SNMP' `configure system > snmp`	Enables and disables SNMP. ■ [0] **No** = (Default) SNMP is enabled.

Parameter	Description
`settings > disable` [DisableSNMP]	■ [1] **Yes** = SNMP is disabled. **Note:** ■ For the parameter to take effect, a device reset is required.
`configure system > snmp settings > port` [SNMPPort]	Defines the device's local (LAN) UDP port used for SNMP Get/Set commands. The range is 100 to 3999. The default port is 161. **Note:** For the parameter to take effect, a device reset is required.
[ChassisPhysicalAlias]	Defines the 'alias' name object for the physical entity as specified by a network manager, and provides a non-volatile 'handle' for the physical entity. The valid range is a string of up to 255 characters.
[ChassisPhysicalAssetID]	Defines the user-assigned asset tracking identifier object for the device's chassis as specified by OVOC, and provides non-volatile storage of this information. The valid range is a string of up to 255 characters.
[ifAlias]	Defines the textual name of the interface. The value is equal to the ifAlias SNMP MIB object. The valid range is a string of up to 64 characters.
`configure system > snmp trap > auto-send-keep-alive` [SendKeepAliveTrap]	Enables the device to send NAT keep-alive traps to the port of the SNMP network management station (e.g., AudioCodes OVOC). This is used for NAT traversal, and allows SNMP communication with OVOC management platform, located in the WAN, when the device is located behind NAT. It is needed to keep the NAT pinhole open for the SNMP messages sent from OVOC to the device. The device sends the

Parameter	Description
	trap periodically - every 9/10 of the time configured by the NATBindingDefaultTimeout parameter. The trap that is sent is acKeepAlive. For more information on the SNMP trap, refer to the *SNMP Reference Guide*. ■ [0] = (Default) Disable ■ [1] = Enable To configure the port number, use the KeepAliveTrapPort parameter. **Note:** For the parameter to take effect, a device reset is required.
[KeepAliveTrapPort]	Defines the port of the SNMP network management station to which the device sends keep-alive traps. The valid range is 0 - 65534. The default is port 162. To enable NAT keep-alive traps, use the [SendKeepAliveTrap] parameter.
[PM_EnableThresholdAlarms]	Enables the sending of the SNMP trap event acPerformanceMonitoringThresholdCrossing, which is sent every time the threshold (high or low) of a Performance Monitored object (e.g., acPMSIPSBCAttemptedCallsTable) is crossed. ■ [0] = (Default) Disable ■ [1] = Enable **Note:** Once enabled, you can change the low and high threshold values per performance monitored object. For more information, see the *SNMP Reference Guide for Gateways-SBCs-MSBRs*.
'Call Duration for Short Calls' `configure voip > sbc settings > short-call-seconds`	Defines the duration (in seconds) of an SBC call for it to be considered as a short call and thus, included in the count of the

Parameter	Description
[ShortCallSeconds]	performance monitoring SNMP MIBs for short calls (acPMSBCInShortCallsTable, acPMSBCOutShortCallsTable, acPMSBCSRDInShortCallsTable, acPMSBCSRDOutShortCallsTable, acPMSBCIPGroupInShortCallsTable, and acPMSBCIPGroupOutShortCallsTable). The valid value is 0 to 60. The default is 2. A value of 0 indicates calls of zero duration, which are calls that do not pass the device's Classification, Manipulation or Routing stages.
`configure system > snmp settings > sys-oid` [SNMPSysOid]	Defines the SNMP MIB OID for the base product system. The default is 1.3.6.1.4.1.5003.8.1.1. **Note:** ■ For the parameter to take effect, a device reset is required. ■ The device automatically adds the device's unique product identifier number at the end of your OID.
[SNMPTrapEnterpriseOid]	Defines the SNMP MIB OID for the Trap Enterprise. The default is 1.3.6.1.4.1.5003.9.10.1.21. **Note:** ■ For the parameter to take effect, a device reset is required. ■ The device automatically adds the device's unique product identifier number at the end of your OID.
[acUserInputAlarmDescription]	Defines the description of the input alarm.
[acUserInputAlarmSeverity]	Defines the severity of the input alarm.
[AlarmHistoryTableMaxSize]	Defines the maximum number of rows in the Alarm History table. The parameter can be controlled by the Config Global Entry Limit MIB (located in the Notification Log

Parameter	Description
	MIB). The valid range is 50 to 1000. The default is 500. **Note:** For the parameter to take effect, a device reset is required.
[ActiveAlarmTableMaxSize]	Defines the maximum number of currently active alarms that can be displayed in the Active Alarms table. When the table reaches this user-defined maximum capacity (i.e., full), the device sends the SNMP trap event, acActiveAlarmTableOverflow. If the table is full and a new alarm is raised by the device, the new alarm is not displayed in the table. The valid range is 100 to 600. The default is 200. For more information on the Active Alarms table, see Viewing Active Alarms. **Note:** ■ For the parameter to take effect, a device reset is required. ■ To clear the acActiveAlarmTableOverflow trap, you must reset the device. The reset also deletes all the alarms in the Active Alarms table.
[ContextEngineID]	Defines the contextEngineID as mentioned in RFC 3411. An SNMP context is a collection of management information accessible by an SNMP entity. An item of management information may exist in more than one context and an SNMP entity potentially has access to many contexts. A context is identified by the snmpEngineID value of the entity hosting the management information (also called a contextEngineID) and a context name that identifies the specific context (also called a contextName).

Parameter	Description
	Note: ■ For the parameter to take effect, a device reset is required. ■ When the device operates in HA mode, this parameter has the same value for both active and redundant devices (i.e., system identifier). If the devices return to Standalone mode (i.e., non-HA mode), you must configure the parameter to a 0 value on both devices. When the devices reset to Standalone mode, each device automatically sets this parameter to a unique value based on its serial number (S/N).
`configure system > snmp settings > engine-id` [SNMPEngineIDString]	Defines the SNMP engine ID for SNMPv2/SNMPv3 agents. This is used for authenticating a user attempting to access the SNMP agent on the device. The ID can be a string of up to 36 characters. The default is 00:00:00:00:00:00:00:00:00:00:00:00 (12 Hex octets characters). The provided key must be set with 12 Hex values delimited by a colon (":") in the format xx:xx:...:xx. For example, 00:11:22:33:44:55:66:77:88:99:aa:bb **Note:** ■ For the parameter to take effect, a device reset is required. ■ Before setting the parameter, all SNMPv3 users must be deleted; otherwise, the parameter setting is ignored. ■ If the supplied key does not pass validation of the 12 Hex values input or it is set with the default value, the engine ID is generated according to RFC 3411.

Parameter	Description
	■ When the device operates in HA mode, the parameter has the same value for both active and redundant devices (i.e., system identifier). If the two devices return to Standalone mode (i.e., non-HA mode), you must configure the parameter to a NULL value (i.e., no value) for both devices. When the devices reset to the Standalone mode, each device automatically sets the parameter to a unique value based on its' serial number (S/N).
Trap Manager Host Name `manager-host-name` [SNMPTrapManagerHostName]	Defines an FQDN of the remote host used as an SNMP manager to receive traps sent by the device. The device sends the traps to the DNS-resolved IP address. The valid range is a string of up to 99 characters. For more information, see Configuring an SNMP Trap Destination with FQDN.
'Activity Trap' `configure troubleshoot > activity-trap` [EnableActivityTrap]	Enables the device to send an SNMP trap to notify of Web user activities in the Web interface. The activities to report are configured by the [ActivityListToLog] parameter (see Reporting Management User Activities on page 1076). ■ [0] **Disable** (default) ■ [1] **Enable**
SNMP Community String Parameters	
'Read Only Community Strings' `configure system > snmp settings > ro-community-string` [SNMPReadOnlyCommunityStringsPassword_x]	Defines a read-only SNMP community string. Up to five read-only community strings can be configured. For more information, see Configuring SNMP Community Strings on page 77. **Note:** ■ The parameter cannot be configured with wide characters.

Parameter	Description
	■ The read-only community strings must be different to the read-write community strings.
'Read/Write Community Strings' `configure system > snmp settings > rw-community-string` [SNMPReadWriteCommunityStringsPassword_x]	Defines a read-write SNMP community string. Up to five read-write community strings can be configured. For more information, see Configuring SNMP Community Strings on page 77. **Note:** ■ The parameter cannot be configured with wide characters. ■ The read-write community strings must be different to the read-only community strings.
'Trap Community String' `configure system > snmp trap > community-string` [SNMPTrapCommunityStringPassword]	Defines the community string for SNMP traps. For more information, see Configuring SNMP Community Strings on page 77. **Note:** The parameter cannot be configured with wide characters.

WebSocket Tunneling with OVOC Parameters

The WebSocket tunneling with OVOC parameters are described in the table below. For more information on WebSocket tunneling between the device and OVOC, see Configuring WebSocket Tunnel with OVOC on page 91.

Table 60-6: WebSocket Tunneling with OVOC Parameters

Parameter	Description
'OVOC WebSocket Tunnel Server Address' `configure network > ovoc-tunnel-settings > address` [WSTunServer]	Defines the address of the WebSocket tunnel server (OVOC). The valid value is an IPv4 address (in dotted-decimal notation) or a hostname. By default, no value is defined. **Note:** ■ For the parameter to take effect, a device reset is required.

Parameter	Description
	■ If you configure the parameter to a hostname, the device uses the DNS server configured in Configuring a DNS Server for HTTP Services on page 343 to resolve it into an IP address.
'Path' `configure network > ovoc-tunnel-settings > path` [WSTunServerPath]	Defines the path of the WebSocket tunnel server. Configure the parameter to "tun" (without quotation marks) to match the default OVOC configuration. **Note:** For the parameter to take effect, a device reset is required.
'Username' `configure network > ovoc-tunnel-settings > username` [WSTunUsername]	Defines the username for connecting the device to the WebSocket tunnel server (OVOC). The valid value is a string of up to 30 characters. Configure the parameter to "VPN" (without quotation marks) to match the default OVOC configuration. **Note:** ■ For the parameter to take effect, a device reset is required. ■ The username must be the same as that configured on OVOC.
'Password' `configure network > ovoc-tunnel-settings > password` [WSTunPassword]	Defines the password for connecting the device to the WebSocket tunnel server (OVOC). The valid value is a string of up to 30 characters. Configure the parameter to "123456" (without quotation marks) to match the default OVOC configuration. **Note:** ■ For the parameter to take effect, a device reset is required.

Parameter	Description
	■ The password must be the same as that configured on OVOC.
'Secured (HTTPS)' `configure network > ovoc-tunnel-settings > secured` [WSTunSecured]	Enables secured (HTTPS) WebSocket tunneling connection. ■ [0] = Disable ■ [1] = (Default) Enable **Note:** For the parameter to take effect, a device reset is required.
'Verify Server' `configure network > ovoc-tunnel-settings > verify-server` [WSTunVerifyPeer]	Enables the device to verify the TLS certificate that is presented by OVOC when establishing the WebSocket tunneling connection. You should load the corresponding CA certificate to the device's Trusted Root store of the default TLS Context (Index #0). ■ [0] = Disable - no certificate verification is done. ■ [1] = (Default) Enable. The device verifies that the TLS certificate presented by OVOC is signed by one of the known CAs. **Note:** ■ For the parameter to take effect, a device reset is required. ■ The parameter is applicable only if you configure the [WSTunSecured] parameter to [1].

Serial Parameters

The serial interface parameters are described in the table below.

Table 60-7: Serial Parameters

Parameter	Description
[DisableRS232]	Enables the device's RS-232 (serial) port.

Parameter	Description
	■ [0] = (Default) Enables RS-232. ■ [1] = Disables RS-232. The RS-232 serial port can be used to change the networking parameters and view error/notification messages. To establish serial communication with the device, see Establishing a CLI Session. **Note:** For the parameter to take effect, a device reset is required.
[SerialBaudRate]	Defines the serial communication baud rate. The valid values include the following: 1200, 2400, 9600, 14400, 19200, 38400, 57600, or 115200 (default). **Note:** For the parameter to take effect, a device reset is required.
[SerialData]	Defines the serial communication data bit. ■ [7] = 7-bit ■ [8] = (Default) 8-bit **Note:** For the parameter to take effect, a device reset is required.
[SerialParity]	Defines the serial communication polarity. ■ [0] = (Default) None ■ [1] = Odd ■ [2] = Even **Note:** For the parameter to take effect, a device reset is required.
[SerialStop]	Defines the serial communication stop bit. ■ [1] = (Default) 1-bit (default) ■ [2] = 2-bit **Note:** For the parameter to take effect, a device reset is required.
[SerialFlowControl]	Defines the serial communication flow control. ■ [0] = (Default) None ■ [1] = Hardware

Parameter	Description
	Note: For the parameter to take effect, a device reset is required.
`configure troubleshoot > startup-n-recovery > startup-dark-mode` [EnableDarkenMode]	Enables serial darkening, which hides the bootup log messages from being displayed in the CLI console during a device reset (boot up). However, if the device fails to load, serial darkening is disabled in the next bootup attempt. ■ [0] ■ [1] (Default) **Note:** For the parameter to take effect, a device reset is required.

Auxiliary and Configuration File Name Parameters

The table below lists the parameters associated with the Auxiliary files. For more information on Auxiliary files, see Loading Auxiliary Files.

Table 60-8: Auxiliary and Configuration File Parameters

Parameter	Description
General Parameters	
[SetDefaultOnIniFileProcess]	Determines if all the device's parameters are set to their defaults before processing the updated *ini* file. ■ [0] = Disable - parameters not included in the downloaded ini file are not returned to default settings (i.e., retain their current settings). ■ [1] = Enable (default). **Note:** The parameter is applicable only for automatic HTTP update or Web *ini* file upload (not applicable if the *ini* file is loaded using BootP).
[SaveConfiguration]	Determines if the device's configuration (parameters and files) is saved to flash (non-volatile memory). ■ [0] = Configuration isn't saved to flash memory. ■ [1] = (Default) Configuration is saved to flash memory.
Auxiliary Filename Parameters	

Parameter	Description
'Call Progress Tones File' [CallProgressTonesFilename]	Defines the name of the file containing the Call Progress Tones definitions. For the ini file, the name must be enclosed by a single quotation mark (e.g., 'cpt_us.dat'). For more information on how to create and load this file, refer to *DConvert Utility User's Guide*. **Note:** For the parameter to take effect, a device reset is required.
'Prerecorded Tones File' [PrerecordedTonesFileName]	Defines the name of the file containing the Prerecorded Tones. **Note:** For the parameter to take effect, a device reset is required.
'Dial Plan File' [DialPlanFileName]	Defines the name of the Dial Plan file. For the ini file, the name must be enclosed by a single quotation mark (e.g., 'dial_plan.dat'). **Note:** This parameter is used only for backward compatibility. For loading (importing) Dial Plan files, use the Dial Plan table instead (see Importing Dial Plans on page 620).
[UserInfoFileName]	Defines the name of the file containing the User Information data. For the ini file, the name must be enclosed by a single quotation mark (e.g., 'userinfo_us.dat'). **Note:** The parameter is only used for backward compatibility.

Automatic Update Parameters

The automatic update of software and configuration files parameters are described in the table below.

Auxiliary file upload through TFTP is not supported by HA mode.

Table 60-9: Automatic Update of Software and Configuration Files Parameters

Parameter	Description
General Automatic Update Parameters `CLI path: configure system > automatic-update`	

Parameter	Description
`update-firmware` [AutoUpdateCmpFile]	Enables the Automatic Update mechanism for the cmp file. ■ [0] = (Default) The Automatic Update mechanism doesn't apply to the cmp file. ■ [1] = The Automatic Update mechanism includes the cmp file. **Note:** For the parameter to take effect, a device reset is required.
`update-frequency-sec` [AutoUpdateFreqencySeconds]	Defines the periodic interval (in seconds) between each automatic update. The valid value range is 0 to 604,800. The default is 0 (i.e., disabled). **Note:** ■ For the parameter to take effect, a device reset is required. ■ This feature can't work with the feature that specifies a specific time of day for automatic updates. Therefore, if you configure this parameter to any value other than 0, leave the [AutoUpdatePredefinedTime] parameter at its default value (i.e., undefined).
`predefined-time` [AutoUpdatePredefinedTime]	Defines the time of day at which the device performs automatic updates. The format syntax of the parameter is 'hh:mm', where *hh* denotes the hour and *mm* the minutes. The value must be enclosed by a single quotation mark (e.g., '20:18'). **Note:** ■ For the parameter to take effect, a device reset is required. ■ By default, the actual update time is randomized by five minutes to reduce the load on the Web servers. To change this randomized time, use the [AutoUpdatePredefinedRandomTime] parameter. ■ This feature can't work with the feature that specifies a periodic interval for automatic

Parameter	Description
	updates. Therefore, if you configure this parameter to any value other than default, leave the [AutoUpdateFreqencySeconds] parameter at its default value (i.e., disabled).
[AutoUpdatePredefinedRandomTime]	Defines the maximum randomization interval (in seconds) for the daily scheduled automatic update (configured by the [AutoUpdatePredefinedTime] parameter). For example, if you configure the [AutoUpdatePredefinedTime] parameter to '13:00' (i.e., 1 pm) and [AutoUpdatePredefinedRandomTime] to '300' (i.e., 5 min.), the actual update can start anywhere between the time 13:00 and 13:05. The valid value range 60 to 86400. The default is 300. **Note:** The parameter is applicable only to the [AutoUpdatePredefinedTime] parameter.
`aupd-graceful-shutdown` [AupdGracefulShutdown]	Enables the device to gracefully lock for the Automatic Update feature when updating the ini configuration file. When the file is downloaded from the provisioning server, the device gracefully locks. During this graceful period (configured by the [AdminStateLockControl] ini file parameter), no new calls are accepted. If all existing calls end before the timeout expires, the device locks and applies the settings of the file. If there are still existing calls when the timeout expires, the device terminates them and applies the settings of the file. For more information, see Applying Downloaded ini File after Graceful Timeout on page 914. ■ [0] = (Default) Disable ■ [1] = Enable
`http-user-agent` [AupdHttpUserAgent]	Defines the information sent in the HTTP User-Agent header in the HTTP Get requests sent by the device to the provisioning server for the Automatic Update mechanism. The valid value is a string of up to 511 characters. The information can include any user-defined string or the following string variable tags (case-sensitive):

Parameter	Description
	■ <NAME>: product name, according to the installed License Key ■ <MAC>: device's MAC address ■ <VER>: software version currently installed on the device, e.g., "7.00.200.001" ■ <CONF>: configuration version, as configured by the ini file parameter, [INIFileVersion] or CLI command `configuration-version` The device automatically populates these tag variables with actual values in the sent header. By default, the device sends the following in the User-Agent header: User-Agent: Mozilla/4.0 (compatible; AudioCodes; <NAME>;<VER>;<MAC>;<CONF>) For example, if you set AupdHttpUserAgent = MyWorld-<NAME>;<VER>(<MAC>), the device sends the following User-Agent header: User-Agent: MyWorld-Mediant;7.00.200.001(00908F1DD0D3) **Note:** ■ The variable tags are case-sensitive. ■ If you configure the parameter with the <CONF> variable tag, you must reset the device with a save-to-flash for your settings to take effect. ■ The tags can be defined in any order. ■ The tags must be defined adjacent to one another (i.e., no spaces).
`auto-firmware` [AutoCmpFileUrl]	Defines the filename and path (URL) to the provisioning server from where the software file (.cmp) can be downloaded, based on timestamp for the Automatic Updated mechanism. The valid value is an IP address in dotted-decimal notation or an FQDN.

Parameter	Description
`aupd-verify-cert` [AUPDVerifyCertificates]	Determines whether the Automatic Update mechanism verifies the TLS certificate received from the provisioning server when the connection is HTTPS. ■ [0] = Disable (default) ■ [1] = Enables TLS certificate verification when the connection with the provisioning server is based on HTTPS. The device verifies the authentication of the certificate received from the provisioning server. The device authenticates the certificate against its trusted root certificate store (see Configuring TLS Certificates on page 158) and if ok, allows communication with the provisioning server. If authentication fails, the device denies communication (i.e., handshake fails).
`credentials` [AUPDUserPassword]	Defines the username and password for digest (MD5 cryptographic hashing) and basic access authentication with the HTTP server on which the files to download are located for the Automatic Update feature. The valid value is a string of up to 128 characters. The syntax is 'username:password' (e.g., 'joe:1234'). By default, no value is defined. **Note:** The device only uses the username and password configured by this parameter if no username and password has been configured for the parameter used to configure the URL of the server with the name of the file, for example, [CmpFileURL].
`crc-check regular` [AUPDCheckIfIniChanged]	Enables the device to perform cyclic redundancy checks (CRC) on downloaded configuration files during the Automatic Update process. The CRC checks whether the content (raw data) of the downloaded file is different to the content of the previously downloaded file from the previous Automatic Update process. The device compares the CRC check value (code) result with the check value of the previously downloaded file. If the check values are identical, it indicates that the file has no

Parameter	Description
	new configuration settings, and the device discards the file. If the check values are different, the device installs the downloaded file and applies the new configuration settings. ■ [0] = (Default) Disable - the device does not perform CRC and installs the downloaded file regardless. ■ [1] = Enable CRC for the entire file, including line order (i.e., same text must be on the same lines). If there are differences between the files, the device installs the downloaded file. If there are no differences, the device discards the newly downloaded file. ■ [2] = Enable CRC for individual lines only. Same as option [1], except that the CRC ignores the order of lines (i.e., same text can be on different lines).
`tftp-block-size` [AUPDTftpBlockSize]	Defines the size of the TFTP data blocks (packets) when downloading a file from a TFTP server for the Automatic Update mechanism. This is in accordance to RFC 2348. TFTP block size is the physical packet size (in bytes) that a network can transmit. When configured to a value higher than the default (512 bytes), but lower than the client network's Maximum Transmission Unit (MTU), the file download speed can be significantly increased. The valid value is 512 to 8192. The default is 512. **Note:** ■ A higher value does not necessarily mean better performance. ■ The block size should be small enough to avoid IP fragmentation in the client network (i.e., below MTU). ■ This feature is applicable only to TFTP servers that support this option.
[ResetNow]	Invokes an immediate device reset. This option can be used to activate offline (i.e., not on-the-fly) parameters that are loaded using the parameter

Parameter	Description
	IniFileUrl. ■ [0] = (Default) The immediate restart mechanism is disabled. ■ [1] = The device immediately resets after an *ini* file with the parameter set to 1 is loaded. **Note:** If you use the parameter in an ini file for periodic automatic provisioning with non-HTTP (e.g., TFTP) and without CRC, the device resets upon every file download.
Software and Configuration File URL Path for Automatic Update Parameters `configure system > automatic-update >`	
`firmware` [CmpFileURL]	Defines the name of the *cmp* file and the URL address (IP address or FQDN) of the server where the file is located. Optionally, the username and password (username:password) for access authentication with the server can also be configured. Example syntax: 'http://192.168.0.1/<filename>' 'https://<username>:<password>@<IP address>/<file name>' **Note:** ■ For the parameter to take effect, a device reset is required. ■ When the parameter is configured, the device always loads the *cmp* file after it is reset. ■ The *cmp* file is validated before it's burned to flash. The checksum of the *cmp* file is also compared to the previously burnt checksum to avoid unnecessary resets. ■ The maximum length of the URL address is 255 characters. ■ When using the ini file, the value must be enclosed by single quotation marks ('...').

Parameter	Description
`ini-file` [IniFileURL]	Defines the name of the *ini* file (configuration) and the URL address (IP address or FQDN) of the server where the file is located. Parameters that are not included in the ini file are restored to default settings. Optionally, the username and password ('https://username:password@10.1.1.1/<file name>') for access authentication with the server can also be configured. Example syntax: 'http://192.168.0.1/<filename>' 'http://192.8.77.13/config_<MAC>.ini' 'https://<username>:<password>@<IP address>/<filename>' **Note:** ■ For the parameter to take effect, a device reset is required. ■ When using HTTP or HTTPS, the date and time of the *ini* file are validated. Only more recently dated *ini* files are loaded. ■ The case-sensitive placeholder "<MAC>" can be used in the filename, and in the names of the folders in the URL path, which is automatically replaced with the device's MAC address. This option allows the loading of specific configurations for specific devices. For more information, see MAC Address Placeholder in Configuration File Name. ■ The maximum length of the URL address is 99 characters. ■ When using the ini file, the value must be enclosed by single quotation marks ('...'). ■ If you want the device to load an ini file where parameters not included in the file remain at their current settings (i.e., incremental), then use the [IncrementalIniFileURL] parameter instead.
`incremental-ini-file`	Defines the name of the incremental *ini* file

Parameter	Description
[IncrementalIniFileURL]	(configuration) and the URL address (IP address or FQDN) of the server where the file is located. Parameters that are not included in the ini file remain at their current settings. Optionally, the username and password ('https://username:password@10.1.1.1/<file name>') for access authentication with the server can also be configured. Example syntax: 'http://192.168.0.1/<filename>' 'http://192.8.77.13/config_<MAC>.ini' 'https://<username>:<password>@<IP address>/<filename>' **Note:** ■ When using HTTP or HTTPS, the date and time of the *ini* file are validated. Only more recently dated *ini* files are loaded. ■ The case-sensitive string "<MAC>" can be used in the filename, and in the names of the folders in the URL path, which is automatically replaced with the device's MAC address. This option allows the loading of specific configurations for specific devices. For more information, see MAC Address Placeholder in Configuration File Name. ■ The maximum length of the URL address is 99 characters. ■ When using the ini file, the value must be enclosed by single quotation marks ('...'). ■ If you want the device to load an ini file where parameters not included in the file are restored to default settings (i.e., not incremental), then use the [IniFileURL] parameter instead.
`cli-script <URL>` [CliScriptURL]	Defines the URL of the server where the CLI Script file containing the device's configuration is located. This file is used for automatic provisioning. Optionally, the username and password ('https://username:password@10.1.1.1/<file name>') for access authentication with the server

Parameter	Description
	can also be configured. **Note:** The case-sensitive string, "<MAC>" can be used in the filename, and in the names of the folders in the URL path, which is automatically replaced with the device's MAC address. For more information, see MAC Address Placeholder in Configuration File Name.
`startup-script <URL>` [CLIStartupScriptUrl]	Defines the URL address of the server where the CLI Startup Script file containing the device's configuration is located. This file is used for automatic provisioning. Optionally, the username and password ('https://username:password@10.1.1.1/<file name>') for access authentication with the server can also be configured. **Note:** ■ The case-sensitive string, "<MAC>" can be used in the file name, and in the names of the folders in the URL path, which is automatically replaced with the device's MAC address. For more information, see MAC Address Placeholder in Configuration File Name. ■ When using the ini file, the value must be enclosed by single quotation marks ('...'). ■ The file is not supported when the device operates in HA mode.
`prerecorded-tones` [PrtFileURL]	Defines the name of the Prerecorded Tones (PRT) file and the URL address (IP address or FQDN) of the server where the file is located. Optionally, the username and password ('https://username:password@10.1.1.1/<file name>') for access authentication with the server can also be configured. Example syntax: 'http://<server_name>/<filename>' 'https://<server_name>/<filename>' **Note:**

Parameter	Description
	■ The maximum length of the URL address is 99 characters. ■ When using the ini file, the value must be enclosed by single quotation marks ('...').
`call-progress-tones` [CptFileURL]	Defines the name of the CPT file and the URL address (IP address or FQDN) of the server where the file is located. Optionally, the username and password ('https://username:password@10.1.1.1/<file name>') for access authentication with the server can also be configured. Example syntax: 'http://<server_name>/<filename>' 'https://<server_name>/<filename>' **Note:** ■ The maximum length of the URL address is 99 characters. ■ When using the ini file, the value must be enclosed by single quotation marks ('...').
`voice-prompts` [VpFileURL]	Defines the name of the Voice Prompts file and the URL address (IP address or FQDN) of the server on which the file is located. Optionally, the username and password ('https://username:password@10.1.1.1/<file name>') for access authentication with the server can also be configured. Example syntax: 'http://<server_name>/<filename>' 'https://<server_name>/<filename>' **Note:** ■ The maximum length of the URL address is 99 characters. ■ When using the ini file, the value must be enclosed by single quotation marks ('...').

Parameter	Description
`dial-plan` [DialPlanCSVFileUrl]	Defines the name of the Dial Plan file (.csv) and the URL address of the server where the file is located. Optionally, the username and password ('https://username:password@10.1.1.1/<filename>') for access authentication with the server can also be configured. **Note:** When using the ini file, the value must be enclosed by single quotation marks ('...').
`tls-root-cert` [TLSRootFileUrl]	Defines the name of the TLS trusted root certificate file and the URL address of the server where the file is located (e.g., tftp://172.17.116.216/Trust.pem). Optionally, the username and password ('https://username:password@10.1.1.1/<filename>') for access authentication with the server can also be configured. **Note:** ■ The parameter replaces all previous loaded trusted root certificate files with the new file. ■ For the parameter to take effect, a device reset is required. ■ When using the ini file, the value must be enclosed by single quotation marks ('...').
`tls-root-cert-incr` [TLSIncrRootFileUrl]	Defines the name of the TLS trusted root certificate file and the URL address of the server where the file is located (e.g., tftp://172.17.116.216/Trust.pem). Optionally, the username and password ('https://username:password@10.1.1.1/<filename>') for access authentication with the server can also be configured. The parameter adds the file to any existing trusted root certificate file (i.e., incremental file load). **Note:** ■ For the parameter to take effect, a device reset is required. ■ When using the ini file, the value must be enclosed by single quotation marks ('...')
`tls-cert` [TLSCertFileUrl]	Defines the name of the TLS certificate file and the URL address of the server where the file is located.

Parameter	Description
	Optionally, the username and password ('https://username:password@10.1.1.1/<file name>') for access authentication with the server can also be configured. **Note:** ■ For the parameter to take effect, a device reset is required. ■ When using the ini file, the value must be enclosed by single quotation marks ('...').
`tls-private-key` [TLSPkeyFileUrl]	Defines the URL address of the server on which the TLS private key file is located. Optionally, the username and password ('https://username:password@10.1.1.1/<file name>') for access authentication with the server can also be configured. **Note:** When using the ini file, the value must be enclosed by single quotation marks ('...').
`sbc-user-info` [SBCUserInfoFileUrl]	Defines the name of the SBC User Information file and the URL address (IP address or FQDN) of the server where the file is located. Optionally, the username and password ('https://username:password@10.1.1.1/<file name>') for access authentication with the server can also be configured. For example: 'https://www.company.com/SBC-User-Info.csv' **Note:** When using the ini file, the value must be enclosed by single quotation marks ('...').
`user-info` [UserInfoFileURL]	Defines the name of the User Information file and the URL address (IP address or FQDN) of the server where the file is located. Optionally, the username and password ('https://username:password@10.1.1.1/<file name>') for access authentication with the server can also be configured. The maximum length of the URL address is 99 characters.

Parameter	Description
	Example syntax: 'http://<server_name>/<filename>' 'https://<server_name>/filename>' **Note:** **Note:** The parameter is used for **backward compatibility only**. Use the [SBCUserInfoFileUrl] parameter (above) instead. ■ When using the ini file, the value must be enclosed by single quotation marks ('...').
`feature-key` [FeatureKeyURL]	Defines the name of the License Key file and the URL address of the server where the file is located. Optionally, the username and password ('https://username:password@10.1.1.1/<filename>') for access authentication with the server can also be configured. **Note:** When using the ini file, the value must be enclosed by single quotation marks ('...').
`template-url` [TemplateUrl]	Defines the URL address in the File Template for automatic updates, of the provisioning server where the files to download are located. Optionally, the username and password ('https://username:password@10.1.1.1/<filename>') for access authentication with the server can also be configured. For more information, see File Template for Automatic Provisioning. **Note:** When using the ini file, the value must be enclosed by single quotation marks ('...').
`template-files-list` [AupdFilesList]	Defines the list of file types in the File Template for automatic updates, to download from the provisioning server. For more information, see File Template for Automatic Provisioning.
`web-favicon` [WebFaviconFileUrl]	Defines the name of the favicon image file and the URL address of the server where the file is located. This is used for the Automatic Update feature.

Parameter	Description
	For more information, see Customizing the Favicon. **Note:** When using the ini file, the value must be enclosed by single quotation marks ('...').
`configuration-pkg` [ConfPackageURL]	Defines the name of the Configuration Package file (.tar.gz) and the URL address (IP address or FQDN) of the server where the file is located. Optionally, the username and password ('https://username:password@10.1.1.1/<file name>') for access authentication with the server can also be configured. For example: `ConfPackageURL = 'http://www.corp.com/ConfBackupPkg5967925.tar.gz'` **Note:** ■ When using the ini file, the value must be enclosed by single quotation marks ('...').

[MatrixCsvFileUrl]	Defines a configuration table as a Comma-Separated Values (CSV) file and the URL address of the server where the file is located.
	The filename must include the name of the configuration table, for example:
	`MatrixCsvFileUrl = 'http://www.corp.com/device_IPGroup.cvs'`
	You can also include in the filename the string variable tag "MAC" (case-sensitive), which the device automatically replaces with its MAC address, for example:
	`MatrixCsvFileUrl = 'http://www.corp.com/device_<MAC>_IPGroup.cvs'`
	This placeholder can also be used in the names of the folders in the URL path, which is automatically replaced with the device's MAC address.
	Note:

Parameter	Description
	■ The parameter is applicable only to tables that support importing CSV files (e.g., Dial Plan table and User Information table). ■ The filename extension must be ".csv". ■ When using the ini file, the value must be enclosed by single quotation marks ('...').
[AUPDResetURLOnWebConfig]	Defines if the URLs configured for the [CmpFileURL] and [IniFileURL] parameters are deleted when you reset the device with a save to flash through the Web interface. ■ [0] = The URLs remain defined for the parameters. ■ [1] = (Default) The URLs are deleted (as the device assumes that you want to manually configure it instead of using the Automatic Update mechanism). **Note:** If you have configured a URL for the [IniFileURL] parameter, the default value of the Web interface's 'Save to Flash' field changes to **No** instead of **Yes** (see Resetting the Device on page 850). This is to make sure that you don't unintentionally save configuration to flash when you reset the device through the Web interface.

Networking Parameters

This subsection describes the device's networking parameters.

Multiple VoIP Network Interfaces and VLAN Parameters

The IP network interfaces and VLAN parameters are described in the table below.

Table 60-10:IP Network Interfaces and VLAN Parameters

Parameter	Description
VLAN Parameters	
[EnableNTPasOAM]	Defines the application type for Network Time Protocol (NTP) services.

Parameter	Description
	■ [1] = OAMP (default) ■ [0] = Control **Note:** For the parameter to take effect, a device reset is required.

Routing Parameters

The IP network routing parameters are described in the table below.

Table 60-11:IP Network Routing Parameters

Parameter	Description
'Don't Send ICMP Unreachable Messages' `configure network > network-settings > icmp-disable-unreachable` [DisableICMPUnreachable]	Defines whether or not the device generates and sends ICMP messages, if required. ■ [0] **Disable** = (Default) Device sends ICMP Unreachable messages. ■ [1] **Enable** = Device does not send ICMP Unreachable messages.
'Send and Receive ICMP Redirect Messages' `configure network > network-settings > icmp-disable-redirec` [DisableICMPRedirects]	Enables sending and receiving ICMP Redirect messages. ■ [0] **Enable** = (Default) Device sends and accepts these messages. ■ [1] **Disable** = Device rejects these messages and also does not send them.

Quality of Service Parameters

The Quality of Service (QoS) parameters are described in the table below.

Table 60-12:QoS Parameters

Parameter	Description
Layer-2 Class Of Service (CoS) Parameters (VLAN Tag Priority Field)	

Parameter	Description
Layer-3 Class of Service (TOS/DiffServ) Parameters `CLI path: configure network > qos application-mapping`	
'Media Premium QoS' `media-qos` [PremiumServiceClassMediaDiffServ]	Global parameter defining the DiffServ value for Premium Media CoS content. You can also configure this feature per specific calls, using IP Profiles (IpProfile_IPDiffServ) . For a detailed description of the parameter and To configure the feature, see Configuring IP Profiles. **Note:** If the feature is configured for a specific profile, the settings of this global parameter is ignored for calls associated with the profile.
'Control Premium QoS' `control-qos` [PremiumServiceClassControlDiffServ]	Global parameter defining the DiffServ value for Premium Control CoS content (Call Control applications). You can also configure the feature per specific calls, using IP Profiles (IpProfile_SigIPDiffServ). For a detailed description of the parameter and To configure the feature in the IP Profiles table, see Configuring IP Profiles. **Note:** If the feature is configured for a specific profile, the settings of this global parameter is ignored for calls associated with the profile.
'Gold QoS' `gold-qos` [GoldServiceClassDiffServ]	Defines the DiffServ value for the Gold CoS content (Streaming applications). The valid range is 0 to 63. The default is 26.
'Bronze QoS' `bronze-qos` [BronzeServiceClassDiffServ]	Defines the DiffServ value for the Bronze CoS content (OAMP applications). The valid range is 0 to 63. The default is 10.

NAT and STUN Parameters

The Network Address Translation (NAT) and Simple Traversal of UDP through NAT (STUN) parameters are described in the table below.

Table 60-13:NAT and STUN Parameters

Parameter	Description
STUN Parameters	
[EnableStunForward]	Enables the device to forward incoming STUN packets (RFC 3849). ■ [0] = (Default) Disable. The device does not forward received STUN packets. ■ [1] = Enable. The device forwards received STUN packets. **Note:** The parameter is applicable only to the SBC application.
NAT Parameters	
'NAT Traversal' `configure voip > media settings > disable-NAT-traversal` [NATMode]	Enables the NAT traversal feature for media when the device communicates with UAs located behind NAT. ■ [0] **Enable NAT Only if Necessary** = NAT traversal is performed only if the UA is located behind NAT: ✓ UA behind NAT: The device sends the media packets to the IP address:port obtained from the source address of the first media packet received from the UA. ✓ UA not behind NAT: The device sends the packets to the IP address:port specified in the SDP 'c=' line (Connection) of the first received SIP message. **Note:** If the SIP session is established (ACK) and the device (not the UA) sends the first packet, it sends it to the address obtained from the SIP message and only after the device receives the first packet from the UA does it determine whether the UA is behind NAT. ■ [1] **Disable NAT** = (Default) The device considers the UA as not located behind NAT and sends media packets to the UA using the IP address:port specified in the SDP 'c=' line (Connection) of the first received SIP message. ■ [2] **Force NAT** = The device always considers the UA as behind NAT and sends the media packets to the IP

Parameter	Description
	address:port obtained from the source address of the first media packet received from the UA. The device only sends packets to the UA after it receives the first packet from the UA (to obtain the IP address). ■ [3] **NAT by Signaling** = The device identifies whether or not the UA is located behind NAT based on SIP signaling. The device assumes that if signaling is behind NAT that the media is also behind NAT, and vice versa. ✓ UA behind NAT: The device sends media according to option **Force NAT** (2). If the 'Media Latch Mode' parameter is configured to **Strict**, the 'Media Latch Mode' parameter automatically changes to **Dynamic**. ✓ UA not behind NAT: The device sends media according to option **Disable NAT** (1). ■ [4] **NAT by Signaling Restricted IP** = The device identifies whether or not the UA is located behind NAT based on SIP signaling. The device assumes that if signaling is behind NAT that the media is also behind NAT, and vice versa. • UA behind NAT: The device sends media only when the source of the media packets is the signaling IP address (source of the INVITE). If the 'Media Latch Mode' parameter is configured to **Strict**, the 'Media Latch Mode' parameter automatically changes to **Dynamic**. • UA not behind NAT: The device sends media according to option **Disable NAT** (1). For more information on NAT traversal, see First Incoming Packet Mechanism.
[NATBindingDefaultTimeout]	The device sends SNMP keep-alive traps periodically - every 9/10 of the time configured by the parameter (in seconds). Therefore, the parameter is applicable only if you configure the [SendKeepAliveTrap] parameter to [1]. The parameter is used to allow SNMP communication with AudioCodes One Voice Operations Center (OVOC)

Parameter	Description
	management platform, located in the WAN, when the device is located behind NAT. It is needed to keep the NAT pinhole open for the SNMP messages sent from OVOC to the device. The valid range is 0 to 2,592,000. The default is 30. **Note:** For the parameter to take effect, a device reset is required.
'SIP NAT Detection' `configure voip > sip-definition advanced-settings > sip-nat-detect` [SIPNatDetection]	Enables the device to detect whether the incoming INVITE message is sent from an endpoint located behind NAT. ■ [0] **Disable** = Disables the device's NAT Detection mechanism. Incoming SIP messages are processed as received from endpoints that are not located behind NAT and sent according to the SIP standard. ■ [1] **Enable** = (Default) Enables the device's NAT Detection mechanism.

DNS Parameters

The Domain name System (DNS) parameters are described in the table below.

Table 60-14:DNS Parameters

Parameter	Description
'Default Primary DNS Server IP' `configure network > dns settings > dns-default-primary-server-ip` [DefaultPrimaryDnsServerIp]	Defines the address of the default primary DNS server. The valid value is an IP address in dotted-decimal notation. The default is 8.8.8.8. For more information, see Configuring Default DNS Servers on page 152.
'Default Secondary DNS Server IP' `configure network > dns settings > dns-default-secondary-server-ip` [DefaultSecondaryDnsServerIp]	Defines the address of the default secondary DNS server. The valid value is an IP address in dotted-

Parameter	Description
	decimal notation. The default is 8.8.4.4. For more information, see Configuring Default DNS Servers on page 152.

DHCP Parameters

The Dynamic Host Control Protocol (DHCP) parameters are described in the table below.

Table 60-15:DHCP Parameters

Parameter	Description
'Enable DHCP' [DHCPEnable]	Enables DHCP client functionality. ■ [0] **Disable** (default) ■ [1] **Enable** **Note:** ■ For the parameter to take effect, a device reset is required. ■ For a detailed description of DHCP, see DHCP-based Provisioning. ■ The parameter is a "hidden" parameter. Once defined and saved to flash memory, its value doesn't revert to default even if the parameter doesn't appear in the *ini* file.
[DhcpOption160Support]	Enables the use of DHCP Option 160. ■ [0] = (Default) Disable ■ [1] = Enable For more information, see Provisioning the Device using DHCP Option 160 on page 905. **Note:** For the parameter to take effect, a device reset is required.
[DHCP120OptionMode]	Enables the acceptance of DHCP Option 120 in DHCP responses sent by a DHCP server. ■ [0] = DHCP Option 120 is not supported and ignored if received in the DHCP response.

Parameter	Description
	■ [1] = (Default) DHCP Option 120 is supported and if received, the device adds the SIP server information to the Proxy Set.
[DHCPSpeedFactor]	Defines the device's DHCP renewal speed for a leased IP address from a DHCP server. ■ [0] = Disable ■ [1] = (Default) Normal ■ [2] to [10] = Fast When set to 0, the DHCP lease renewal is disabled. Otherwise, the renewal time is divided by this factor. Some DHCP-enabled routers perform better when set to 4. **Note:** For the parameter to take effect, a device reset is required.

Clock (Date and Time) Synchronization Parameters

The device's clock synchronization parameters are described in the table below.

Table 60-16:Device Clock Synchronization Parameters

Parameter	Description
NTP `CLI path: configure system > ntp` **Note:** For more information on Network Time Protocol (NTP), see Simple Network Time Protocol Support.	
'Primary NTP Server Address' `primary-server` [NTPServerIP]	Defines the IP address (in dotted-decimal notation or as an FQDN) of the NTP server. The advantage of using an FQDN is that multiple IP addresses can be resolved from the DNS server, providing NTP server redundancy. The default IP address is 0.0.0.0 (i.e., internal NTP client is disabled).
'Secondary NTP Server Address' `secondary-server` [NTPSecondaryServerIP]	Defines a second NTP server's address as an FQDN or an IP address (in dotted-decimal notation). This NTP is used for redundancy; if the primary NTP server fails, then this NTP server is used. The default IP address is 0.0.0.0.

Parameter	Description
'NTP Update Interval' `update-interval` [NTPUpdateInterval]	Defines the time interval (in seconds) that the NTP client requests for a time update. The default interval is 86400 (i.e., 24 hours). The range is 0 to 214783647. **Note:** It is not recommend to set the parameter to beyond one month (i.e., 2592000 seconds).
'NTP Authentication Key Identifier' `auth-key-id` [NtpAuthKeyId]	Defines the NTP authentication key identifier for authenticating NTP messages. The identifier must match the value configured on the NTP server. The NTP server may have several keys configured for different clients; this number identifies which key is used. The valid value is 1 to 65535. The default is 0 (i.e., no authentication is done).
'NTP Authentication Secret Key' `auth-key-md5` [ntpAuthMd5KeyPassword]	Defines the secret authentication key shared between the device (client) and the NTP server for authenticating NTP messages. The valid value is a string of up to 32 characters. By default, no key is defined. **Note:** The parameter cannot be configured with wide characters.
Regional Clock and Daylight Saving Time	
'UTC Offset' `configure system > clock > utc-offset` [NTPServerUTCOffset]	Defines the Universal Time Coordinate (UTC) offset (in seconds) from the local time. The valid range is -86400 seconds (i.e., -24 hours) to +86400 seconds (i.e., +24 hours). The default is 0. **Note:** The offset setting is applied only on the hour. For example, if you configure the parameter at 15:42, the device applies the setting only at 16:00.
'Daylight Saving Time' `configure system > clock > summer-time > summer-time` [DayLightSavingTimeEnable]	Enables daylight saving time (DST). ■ [0] **Disable** (default) ■ [1] **Enable**
'Start Time / Day of Month Start' `configure system > clock > summer-time >`	Defines the date and time when DST begins. This value can be configured using any of the following formats:

Parameter	Description
`start` [DayLightSavingTimeStart]	■ Day of year - *mm:dd:hh:mm*, where: ✓ *mm* denotes month ✓ *dd* denotes date of the month ✓ *hh* denotes hour ✓ *mm* denotes minutes For example, "05:01:08:00" denotes daylight saving starting from May 1 at 8 A.M. ■ Day of month - *mm:day/wk:hh:mm*, where: ✓ *mm* denotes month (e.g., 04) ✓ *day* denotes day of week (e.g., FRI) ✓ *wk* denotes week of the month (e.g., 03) ✓ *hh* denotes hour (e.g., 23) ✓ *mm* denotes minutes (e.g., 10) For example, "04:FRI/03:23:00" denotes Friday, the third week of April, at 11 P.M. The week field can be 1-5, where 5 denotes the last occurrence of the specified day in the specified month. For example, "04:FRI/05:23:00" denotes the last Friday of April, at 11 P.M.
'End Time / Day of Month End' `configure system > clock > summer-time > end` [DayLightSavingTimeEnd]	Defines the date and time when DST ends. For a description of the format of this value, see the DayLightSavingTimeStart parameter.
'Offset' `configure system > clock > summer-time > offset` [DayLightSavingTimeOffset]	Defines the DST offset (in minutes). The valid range is 0 to 120. The default is 60. **Note:** The offset setting is applied only on the hour. For example, if you configure the parameter at 15:42, the device applies the setting only at 16:00.
Date Header Date and Time Synchronization	
'Synchronize Time from SIP Date Header' `date-header-time-sync`	Enables the device to obtain its date and time for its internal clock from the SIP Date header in 200 OK messages received in response to sent REGISTER messages.

Parameter	Description
[DateHeaderTimeSync]	■ [0] **Disable** (default) ■ [1] **Enable** For more information, see Configuring Automatic Date and Time through SIP on page 105.
'Time Synchronization Interval' `date-header-time-sync-interval` [DateHeaderTimeSyncInterval]	Defines the minimum time (in seconds) between synchronization updates using the SIP Date header method for clock synchronization. The valid value range is 60 to 86,400. The default is 900. For more information, see Configuring Automatic Date and Time through SIP on page 105.

Debugging and Diagnostics Parameters

This subsection describes the device's debugging and diagnostic parameters.

General Parameters

The general debugging and diagnostic parameters are described in the table below.

Table 60-17:General Debugging and Diagnostic Parameters

Parameter	Description
'Delay After Reset [sec]' `configure voip > sip-definition advanced-settings > delay-after-reset` [GWAppDelayTime]	Defines the time interval (in seconds) that the device's operation is delayed after a reset. The valid range is 0 to 45. The default is 7 seconds. **Note:** This feature helps overcome connection problems caused by some LAN routers or IP configuration parameters' modifications by a DHCP server.
[EnableAutoRAITransmitBER]	Enables the device to send a remote alarm indication (RAI) when the bit error rate (BER) is greater than 0.001. ■ [0] = Disable (default) ■ [1] = Enable
[DspFarmsInstalledNum]	Defines the number of MPM modules installed in the device's chassis. The parameter is typically used for debugging the MPM modules. If the number of configured MPM modules by this parameter is greater

Parameter	Description
	than the number of physical MPM modules detected by the device (i.e., installed in the chassis), the device sends the AcDSPFarmsMismatchAlarm SNMP alarm. For more information on this alarm, refer to the SNMP Reference Guide for SBC-Gateway-MSBR. The valid value is 0 to 3. The default is 0. **Note:** For the parameter to take effect, a device reset is required.

SIP Test Call Parameters

The SIP Signaling Test Call parameters are described in the table below.

Table 60-18:SIP Test Call Parameters

Parameter	Description
'Test Call DTMF String' `configure troubleshoot > test-call settings > testcall-dtmf-string` [TestCallDtmfString]	Defines the DTMF tone that is played for answered test calls (incoming and outgoing). The DTMF string can be up to 15 strings. The default is "3212333". If no string is defined (empty), DTMF is not played.
'Test Call ID' `configure troubleshoot > test-call settings > testcall-id` [TestCallID]	Defines the test call prefix number (*ID*) of the simulated phone on the device. Incoming calls received with this called prefix number are identified as test calls. This can be any string of up to 15 characters. By default, no number is defined. **Note:** The parameter is only for testing incoming calls destined to this prefix number.

Syslog, CDR and Debug Parameters

The Syslog, CDR and debug parameters are described in the table below.

Table 60-19:Syslog, CDR and Debug Parameters

Parameter	Description
'Enable Syslog' `configure troubleshoot > syslog > syslog`	Determines whether the device sends logs and error messages (e.g., CDRs) generated by the device to a Syslog server.

Parameter	Description
[EnableSyslog]	■ [0] **Disable** (default) ■ [1] **Enable** **Note:** ■ If you enable Syslog, you must configure the address of the Syslog server, using the [SyslogServerIP] parameter. ■ Syslog messages may increase the network traffic. ■ To configure Syslog SIP message logging levels, use the [GwDebugLevel] parameter.
'Syslog Server IP' `configure troubleshoot > syslog > syslog-ip` [SyslogServerIP]	Defines the IP address (in dotted-decimal notation) of the computer on which the Syslog server is running. The Syslog server is an application designed to collect the logs and error messages generated by the device. The default IP address is 0.0.0.0.
'Syslog Server Port' `configure troubleshoot > syslog > syslog-port` [SyslogServerPort]	Defines the UDP port of the Syslog server. The valid range is 0 to 65,535. The default port is 514.
`configure troubleshoot > syslog > network-source` [SyslogInterface_InterfaceName]	
'Log Severity Level' `log-level` [SyslogLogLevel]	Defines the minimum severity level of messages included in the Syslog message that is generated by the device. The specified severity level and all higher severity levels are included in the Syslog message. For example, if you configure the parameter to **Alert**, the Syslog will include messages with **Alert** severity level and messages with **Fatal** severity level. The severity levels are listed below from highest to lowest.

Parameter	Description
	■ [0] **Fatal** ■ [1] **Alert** ■ [2] **Critical** ■ [3] **Error** ■ [4] **Warning** ■ [5] **Notice** (default) ■ [6] **Info [not recommended]** ■ [7] **Debug [not recommended]** **Note:** It's **strongly recommended** to leave the Syslog severity level at its default setting. Changing severity level is typically done only by AudioCodes Support for debugging.
'CDR Syslog Server IP Address' `configure troubleshoot > cdr > cdr-srvr-ip-adrr` [CDRSyslogServerIP]	Defines the destination address (IP address) to where CDR logs are sent. The default value is a null string, which causes CDR messages to be sent with all Syslog messages to the Syslog server. **Note:** ■ The CDR messages are sent to UDP port 514 (default Syslog port). ■ This mechanism is active only when Syslog is enabled (i.e., the parameter [EnableSyslog] is set to [1]).
'Call-End CDR SIP Reasons Filter' `configure troubleshoot > cdr > call-end-cdr-sip-reasons-filter` [CallEndCDRSIPReasonsFilter]	Defines SIP release cause codes that if received for the call, the devicedoes not sent Call-End CDRs for the call. The valid value is 300 through to 699. You can configure the parameter with multiple codes using a comma to separate them (e.g., 301,400,404). You can also use "xx" to denote a range (e.g., 3xx).
'Call-End CDR Zero Duration Filter' `configure troubleshoot > cdr > call-end-cdr-zero-duration-`	Enables the device to not send Call-End CDRs if the call's duration is zero (0). ■ [0] Disable (default)

Parameter	Description
`filter` [CallEndCDRZeroDurationFilter]	■ [1] Enable
'CDR Report Level' `configure troubleshoot > cdr > cdr-report-level` [CDRReportLevel]	Enables signaling-related CDRs to be sent to a Syslog server and defines the call stage at which they are sent. ■ [0] **None** = (Default) CDRs are not used. ■ [1] **End Call** = CDR is sent to the Syslog server at the end of each call. ■ [2] **Start & End Call** = CDR report is sent to Syslog at the start and end of each call. ■ [3] **Connect & End Call** = CDR report is sent to Syslog at connection and at the end of each call. ■ [4] **Start & End & Connect Call** = CDR report is sent to Syslog at the start, at connection, and at the end of each call. **Note:** ■ The parameter enables only signaling-related CDRs. To enable media-related CDRs for SBC calls, use the [MediaCDRReportLevel] parameter. ■ The CDR Syslog message complies with RFC 3164 and is identified by: Facility = 17 (local1) and Severity = 6 (Informational). ■ This mechanism is active only when Syslog is enabled (i.e., the parameter [EnableSyslog] is set to [1]).
'Media CDR Report Level' `configure troubleshoot > cdr > media-cdr-rprt-level` [MediaCDRReportLevel]	Enables media-related CDRs of SBC calls to be sent to a Syslog server and defines the call stage at which they are sent. ■ [0] **None** = (Default) No media-related CDR is sent. ■ [1] **End Media** = Sends a CDR only at the end of the call.

Parameter	Description
	■ [2] **Start & End Media** = Sends a CDR once the media starts. In some calls it may only be after the call is established, but in other calls the media may start at ringback tone. A CDR is also sent upon termination (end) of the media in the call. ■ [3] **Update & End Media** = Sends a CDR when an update occurs in the media of the call. For example, a call starts and a ringback tone occurs, a re-INVITE is sent for a fax call and as a result, a CDR with the MediaReportType field set to "Update" is sent, as the media was changed from voice to T.38. A CDR is also sent upon termination (end) of the media in the call. ■ [4] **Start & End & Update Media =** Sends a CDR at the start of the media, upon an update in the media (if occurs), and at the end of the media. **Note:** ■ To enable CDR generation as well as enable signaling-related CDRs, use the CDRReportLevel parameter.
'REST CDR Report Level' `configure system > cdr > rest-cdr-report-level` [RestCdrReportLevel]	Enables signaling-related CDRs to be sent to a REST server and defines the call stage at which they are sent. ■ [0] **None** = (Default) CDRs are not sent. ■ [1] **End Call** = CDRs are sent at the end (SIP BYE) of each call. ■ [2] **Start & End Call** = CDRs are sent at the start (SIP INVITE) and end of each call. ■ [3] **Connect & End Call** = CDRs are sent at call connection (200 OK) and end of each call. ■ [4] **Start & End & Connect Call** = CDRs

Parameter	Description
	are sent at the start, connection, and end of each call. ■ [5] **Connect Only** = CDRs are sent at call connection. **Note:** ■ To specify the REST server, use the [RestCdrHttpServer] parameter. ■ For the device to generate CDRs, you must enable Syslog messaging (see the [EnableSyslog] parameter). ■ CDRs are sent in JSON format.
'REST CDR HTTP Server Name' `configure system > cdr > rest-cdr-http-server` [RestCdrHttpServer]	Defines the REST server (by name as configured in the Remote Web Services table) to where the device sends CDRs through REST API. The valid value is a string (i.e., name of the REST server). By default, no value is defined. **Note:** ■ The parameter value is case sensitive. ■ To enable CDR generation for the REST server, see the [RestCdrReportLevel] parameter. ■ The REST server is configured in the Remote Web Services table (see Configuring Remote Web Services on page 316).
`configure troubleshoot > cdr > cdr-history-privacy` [CDRHistoryPrivacy]	Enables the device to hide the values of the Caller and Callee fields in CDRs of certain report outputs. ■ [0] = (Default) Field values are shown. ■ [1] = Field values are hidden (replaced by an * asterisk). For more information, see Hiding Caller and Callee CDR Field Values on page 1046.

Parameter	Description
'Call Success SIP Reasons' `configure troubleshoot > cdr > call-success-sip-reasons` [CallSuccessSIPReasons]	Defines the SIP response code that you want the device to consider as call success, which is indicated by the optional 'Call Success' field in the sent CDR. This parameter overrides the device's default behavior of how it considers calls a success or failure based on SIP responses. The valid value is string of up to 128 characters to represent SIP response codes (e.g., 486). You can configure the parameter with multiple response codes, whereby each code is separated by a comma without spaces before or after (e.g., 486,408,406). You can also configure a range of responses using the "xx" wildcard (e.g., 4xx,502). By default, no value is defined. **Note:** If an overlap of a SIP response occurs between the configured 'Call Success SIP Reasons' and 'Call Failure SIP Reasons' parameters, the device uses the parameter that is configured with the specific response code, instead of the parameter configured with the range ("xx"). For example, if you configure the 'Call Success SIP Reasons' parameter with "486,5xx" and the 'Call Failure SIP Reasons' parameter with "502", for 502 responses, the device uses the settings of the 'Call Failure SIP Reasons' parameter only. In other words, a call with SIP response code 502 is considered as a call failure.
'Call Failure SIP Reasons' `call-failure-sip-reasons` [CallFailureSIPReasons]	Defines the SIP response codes that you want the device to consider as call failure, which is indicated by the optional 'Call Success' field in the sent CDR. This parameter overrides the device's default behavior of how it considers calls a success or failure based on SIP responses. The valid value is string of up to 128 characters to represent SIP response codes (e.g., 486). You can configure the parameter

Parameter	Description
	with multiple response codes, whereby each code is separated by a comma without spaces before or after (e.g., 486,408,406). You can also configure a range of responses using the "xx" wildcard (e.g., 4xx,502). By default, no value is defined. **Note:** If an overlap of a SIP response occurs between the configured 'Call Success SIP Reasons' and 'Call Failure SIP Reasons' parameters, the device uses the parameter that is configured with the specific response code, instead of the parameter configured with the range ("xx"). For example, if you configure the 'Call Success SIP Reasons' parameter with "486,5xx" and the 'Call Failure SIP Reasons' parameter with "502", for 502 responses, the device uses the settings of the 'Call Failure SIP Reasons' parameter only. In other words, a call with SIP response code 502 is considered as a call failure.
'Call Success Internal Reasons' `call-success-internal-reasons` [CallSuccessInternalReasons]	Defines the internal response codes (generated by the device) that you want the device to consider as call success, which is indicated by the optional 'Call Success' field in the sent CDR. This parameter overrides the device's default behavior of how it considers calls a success or failure based on internally responses. The valid value is string of up to 128 characters to represent internal response codes (e.g., 851). You can configure the parameter with multiple response codes, whereby each code is separated by a comma without spaces before or after (e.g., 851,320). You can also configure a range of responses using the "xx" wildcard (e.g., 8xx,320). By default, no value is defined. **Note:** ■ For a list of the internal response codes, see the 'Termination Reason' [410] CDR

Parameter	Description
	field in CDR Field Description on page 990. ■ If an overlap of a SIP response occurs between the configured 'Call Success SIP Reasons' and 'Call Failure SIP Reasons' parameters, the device uses the parameter that is configured with the specific response code, instead of the parameter configured with the range ("xx"). For example, if you configure the 'Call Success SIP Reasons' parameter with "320,8xx" and the 'Call Failure SIP Reasons' parameter with "851", for 851 responses, the device uses the settings of the 'Call Failure SIP Reasons' parameter only. In other words, a call with response code 851 is considered as a call failure.
'Call Failure Internal Reasons' `call-failure-internal-reasons` [CallFailureInternalReasons]	Defines the internal response codes (generated by the device) that you want the device to consider as call failure, which is indicated by the optional 'Call Success' field in the sent CDR. This parameter overrides the device's default behavior of how it considers calls a success or failure based on internally responses. The valid value is string of up to 128 characters to represent internal response codes (e.g., 851). You can configure the parameter with multiple response codes, whereby each code is separated by a comma without spaces before or after (e.g., 851,320). You can also configure a range of responses using the "xx" wildcard (e.g., 8xx,320). By default, no value is defined. **Note:** ■ For a list of the internal response codes, see the 'Termination Reason' [410] CDR field in CDR Field Description on page 990.

Parameter	Description
	■ If an overlap of a SIP response occurs between the configured 'Call Success SIP Reasons' and 'Call Failure SIP Reasons' parameters, the device uses the parameter that is configured with the specific response code, instead of the parameter configured with the range ("xx"). For example, if you configure the 'Call Success SIP Reasons' parameter with "320,8xx" and the 'Call Failure SIP Reasons' parameter with "851", for 851 responses, the device uses the settings of the 'Call Failure SIP Reasons' parameter only. In other words, a call with response code 851 is considered as a call failure.
'No User Response Before Connect' `no-user-response-before-connect` [NoUserResponseBeforeConnectSuccess]	Defines if the device considers a call as a success or failure when the internal response (generated by the device) "GWAPP_NO_USER_RESPONDING" (18) is received before call connect (SIP 200 OK). ■ [0] **Call Failure** ■ [1] **Call Success** (default)
'No User Response After Connect' `no-user-response-after-connect` [NoUserResponseAfterConnectSuccess]	Defines if the device considers a call as a success or failure when the internal response (generated by the device) "GWAPP_NO_USER_RESPONDING" (18) is received after call connect (SIP 200 OK). ■ [0] **Call Failure** (default) ■ [1] **Call Success**
'Call Transferred before Connect' `call-transferred-before-connect` [CallTransferredBeforeConnectSuccess]	Defines if the device considers a call as a success or failure when the internal response (generated by the device) "RELEASE_BECAUSE_CALL_TRANSFERRED" (807) is generated before call connect (SIP 200 OK). ■ [0] **Call Failure** (default) ■ [1] **Call Success**

Parameter	Description
'Call Transferred after Connect' `call-transferred-after-connect` [CallTransferredAfterConnectSuccess]	Defines if the device considers a call as a success or failure when the internal response (generated by the device) "RELEASE_BECAUSE_CALL_TRANSFERRED" (807) is generated after call connect (SIP 200 OK). ■ [0] **Call Failure** ■ [1] **Call Success** (default)
'File Size' `configure troubleshoot > cdr > file-size` [CDRLocalMaxFileSize]	Defines the size (in kilobytes) of each locally stored CDR file. When the Current file reaches this size, the device creates a CDR file containing all the CDRs from the Current file. The valid value is 1024 to 10,000. The default is 1024. **Note:** ■ CDR file creation works together with the 'Rotation Period' parameter, whereby the file is created as soon as one of the parameter's ('File Size' or 'Rotation Period') settings are fulfilled (whichever is met earlier). For example, if the 'File Size' parameter is 100 and 'Rotation Period' is 60, and the file size reaches 100 kbytes after only 30 minutes has passed, the device creates the CDR file. ■ The parameter is applicable only to local storage of CDRs.
'Number of Files' `configure troubleshoot > cdr > files-num` [CDRLocalMaxNumOfFiles]	Defines the maximum number of locally stored CDR files. If the maximum number is reached and a new file is created, the oldest file is deleted to make space for the new file (i.e., FIFO). The valid value is 2 to 4096 . The default is 5. **Note:** The parameter is applicable only to local storage of CDRs.

Parameter	Description
'Rotation Period' `configure troubleshoot > cdr > rotation-period` [CDRLocalInterval]	Defines how often (in minutes) the device creates a new CDR file for locally stored CDRs. For example, if configured to 60, every hour it creates a CDR file containing all the CDRs from the Current file. The valid value is 2 to 1440. The default is 60. **Note:** ■ CDR file creation works together with the 'File Size' parameter, whereby the file is created as soon as one of the parameter's ('File Size' or 'Rotation Period') settings are fulfilled (whichever is met earlier). For example, if the 'Rotation Period' parameter is 60 and 'File Size' is 100, and an hour has passed but the file size is only 50 kbytes, the device creates the CDR file. ■ The CDR file is created even if there are no CDRs in the Current file. ■ The parameter is applicable only to local storage of CDRs.
'VoIP Debug Level' `configure troubleshoot > syslog > debug-level` [GwDebugLevel]	Enables Syslog debug reporting and logging level. ■ [0] **No Debug** = (Default) Debug is disabled and Syslog messages are not sent. ■ [1] **Basic** = Sends debug logs of incoming and outgoing SIP messages. ■ [5] **Detailed** = Sends debug logs of incoming and outgoing SIP message as well as many other logged processes.
`configure system > cdr > non-call-cdr-rprt` [EnableNonCallCdr]	Enables creation of CDR messages for non-call SIP dialogs (such as SUBSCRIBE, OPTIONS, and REGISTER). ■ [0] = (Default) Disable ■ [1] = Enable

Parameter	Description
'Syslog Optimization' `configure troubleshoot > syslog > syslog-optimization` [SyslogOptimization]	Enables the device to accumulate and bundle multiple debug messages into a single UDP packet and then send it to a Syslog server. The benefit of this feature is that it reduces the number of UDP Syslog packets, thereby improving (optimizing) CPU utilization. ■ [0] **Disable** (default) ■ [1] **Enable** **Note:** The size of the bundled message is configured by the MaxBundleSyslogLength parameter.
`mx-syslog-lgth` [MaxBundleSyslogLength]	Defines the maximum size (in bytes) threshold of logged Syslog messages bundled into a single UDP packet, after which they are sent to a Syslog server. The valid value range is 0 to 1220 (where 0 indicates that no bundling occurs). The default is 1220. **Note:** The parameter is applicable only if the [GWDebugLevel] parameter is enabled.
'Syslog CPU Protection' `configure troubleshoot > syslog > syslog-cpu-protection` [SyslogCpuProtection]	Enables the protection of the device's CPU resources during debug reporting, ensuring voice traffic is unaffected. If CPU resources drop (i.e., high CPU usage) to a critical level (threshold), the device automatically lowers the debug level to free up CPU resources that were required for the previous debug-level functionality. When sufficient CPU resources become available again, the device increases the debug level. The threshold is configured by the 'Debug Level High Threshold' parameter (see below). ■ [0] **Disable** ■ [1] **Enable** (default)
'Debug Level High Threshold' `configure troubleshoot > syslog > debug-level-high-`	Defines the threshold (in percentage) for automatically switching to a different debug level, depending on CPU usage. The

Parameter	Description
`threshold` [DebugLevelHighThreshold]	parameter is applicable only if the 'Syslog CPU Protection' parameter is enabled. The valid value is 0 to 100. The default is 90. The debug level is changed upon the following scenarios: ■ CPU usage equals threshold: Debug level is reduced one level. ■ CPU usage is at least 5% greater than threshold: Debug level is reduced another level. ■ CPU usage is 5 to 19% less than threshold: Debug level is increased by one level. ■ CPU usage is at least 20% less than threshold: Debug level is increased by another level. For example, assume that the threshold is set to 70% and the Debug Level to Detailed (5). When CPU usage reaches 70%, the debug level is reduced to Basic (1). When CPU usage increases by 5% or more than the threshold (i.e., greater than 75%), the debug level is disabled - No Debug (0). When the CPU usage decreases to 5% less than the threshold (e.g., 65%), the debug level is increased to Basic (1). When the CPU usage decreases to 20% less than the threshold (e.g., 50%), the debug level changes to Detailed (5). **Note:** The device does not increase the debug level to a level that is higher than what you configured for the 'Debug Level' parameter.
`configure troubleshoot > cdr > time-zone-format` [TimeZoneFormat]	Defines the time zone that is displayed with the timestamp in CDRs. The timestamp appears in the CDR fields "Setup Time", "Connect Time", and "Release Time". The valid value is a string of up to six characters. The default is UTC. For example,

Parameter	Description
	if you configure the parameter TimeZoneFormat = GMT+11, the timestamp in CDRs are generated with the following time zone display: 17:47:45.411 **GMT+11** Sun Jan 03 2018 **Note:** The time zone is only for display purposes; it does not configure the actual time zone.
`configure troubleshoot > cdr > call-duration-units` [CallDurationUnits]	Defines the unit of measurement for call duration ("Duration" field) in CDRs generated by the device. ■ [0] Seconds (default) ■ [1] Deciseconds ■ [2] Centiseconds ■ [3] Milliseconds The parameter applies to CDRs for Syslog, RADIUS, local-device storage, and CDR history displayed in the Web interface.
'CDR Syslog Sequence Number' `configure system > cdr > cdr-seq-num` [CDRSyslogSeqNum]	Enables or disables the inclusion of the sequence number (S=) in CDR Syslog messages. ■ [0] **Disable** ■ [1] **Enable** (default)
[SendAcSessionIDHeader]	Enables the use of the Global Session ID in SIP messages (AC-Session-ID header), which is a unique identifier of the call session, even if it traverses multiple devices. ■ [0] = (Default) Disables the feature. The device sends outgoing SIP messages without a Global Session ID (even if a Global Session ID was received in the incoming SIP message). ■ [1] = Enables the feature. If the device receives an incoming SIP message containing a Global Session ID, it sends the same Global Session ID in the

Parameter	Description
	outgoing SIP message. If the incoming SIP message does not contain a Global Session ID or if a new session is initiated by the device, the device generates a new, unique Global Session ID and adds it to the outgoing SIP message. For more information, see Enabling Same Call Session ID over Multiple Devices on page 1099.
'Activity Types to Report via Activity Log Messages' `configure troubleshoot > activity-log` [ActivityListToLog]	Defines the operations (activities) performed in the Web interface that are reported to a Syslog server. ■ [pvc] **Parameters Value Change** = Changes made on-the-fly to parameters and tables, and Configuration file load. Note that the ini file parameter, EnableParametersMonitoring can also be used to set this option. ■ [afl] **Auxiliary Files Loading** = Loading of Auxiliary files. ■ [dr] **Device Reset** = Resetting the device from the Maintenance Actions page. Note: For this option to take effect, a device reset is required. ■ [fb] **Flash Memory Burning** = Saving configuration with burn to flash from the Maintenance Actions page. ■ [swu] **Device Software Update** = Software updates (i.e., loading of cmp file) through the Software Upgrade Wizard. ■ [naa] **Non-Authorized Access** = Attempts to log in to the Web interface with a false or empty username or password. ■ [spc] **Sensitive Parameters Value Change** = Changes made to "sensitive" parameters:

Parameter	Description
	✓ (1) IP Address ✓ (2) Subnet Mask ✓ (3) Default Gateway IP Address ✓ (4) ActivityListToLog ■ [ll] **Login and Logout** = Web login and logout attempts. ■ [cli] **CLI Activity** = CLI commands entered by the user. ■ [ae] **Action Executed** = Logs user actions that are not related to parameter changes. The actions can include, for example, file uploads, file delete, lock-unlock maintenance actions, LDAP clear cache, register-unregister, and start-stop trunk. In the Web, these actions are typically done by clicking a button (e.g., the **LOCK** button). **Note:** For the *ini* file parameter, enclose values in single quotation marks, for example: ActivityListToLog = 'pvc', 'afl', 'dr', 'fb', 'swu', 'naa', 'spc'.
[EnableParametersMonitoring]	Enables the monitoring, through Syslog messages, of parameters that are modified on-the-fly. ■ [0] = (Default) Disable ■ [1] = Enable
'Debug Recording Destination IP' `configure troubleshoot > logging settings > dbg-rec-dest-ip` [DebugRecordingDestIP]	Defines the IP address (IPv4) of the server for capturing debug recording. For more information, see Configuring the Debug Recording Server Address on page 1085.
'Debug Recording Destination Port' `configure troubleshoot > logging settings > dbg-rec-dest-port` [DebugRecordingDestPort]	Defines the UDP port of the server for capturing debug recording. The default is 925.

Parameter	Description
'Enable Core Dump' [EnableCoreDump]	Enables the automatic generation of a Core Dump file upon a device crash. ■ [0] **Disable** (default) ■ [1] **Enable** **Note:** For the parameter to take effect, a device reset is required.
'Core Dump Destination IP' [CoreDumpDestIP]	Defines the IP address of the remote server where you want the device to send the Core Dump file. By default, no IP address is defined.
'Call Flow Report Mode' `call-flow-report` [CallFlowReportMode]	Enables the device to send SIP call messages to OVOC so that OVOC can display SIP call dialog sessions as SIP call flow diagrams. ■ [0] **Disable** (default) ■ [1] **Enable** For more information, see Enabling SIP Call Flow Diagrams in OVOC on page 1097.
`configure troubleshoot > syslog > system-log-size` [SystemLogSize]	Defines the size (in Kbytes) of the system log file. The valid value range is 10 to 2000 KB. The default is 200 KB. To view the logged information in this file, use the CLI command `show system log`.
[PLThresholdLevelsPerMille]	Defines packet-loss percentage ranges that are used in sent Syslog messages to report packet loss in incoming media streams (RTP) in 15-second intervals. The valid value range is 1 to 1,000. The default is 5, 10, 20, 50. The syntax for configuring the parameter is: PLThresholdLevelsPerMille = Level1, Level2, Level3, Level4 Where the levels represent the following ranges in the Syslog: ■ [No PL]

Parameter	Description
	■ [up to (Level1/10)%] ■ [(Level1/10)% - (Level2/10)%] ■ [(Level2/10)% - (Level3/10)%] ■ [(Level3/10)% - (Level4/10)%] ■ [(Level4/10)% - 100%] For example (using default values): `PLThresholdLevelsPerMille = 5,10,20,50` Therefore, the ranges are: ■ [No PL] ■ [up to 0.5%] ■ [0.5% - 1%] ■ [1% - 2%] ■ [2% - 5%] ■ [5% - 100%] For more information, see Packet Loss Indication in Syslog on page 1083.

Heartbeat Packet Parameters

The Heartbeat packet parameters are described in the table below. The device sends a heartbeat packet to ensure that the far-end is passing traffic.

Table 60-20:Heartbeat Packet Parameters

Parameter	Description
[HeartBeatIntervalmsec]	Defines the delay (in msec) between consecutive heartbeat packets. The parameter is used when the device operates in High-Availability (HA) mode. The active and redundant devices send heartbeat (keep-alive) messages to one another to check reachability. They send the keep-alive messages "in bulks" (from multiple local ports) every 100 msec to create multiple flows in the network equipment (e.g., switches or SDN network). If there is no reply from the active device to the sent heartbeat packets within this timeout (HA timeout), the active device is considered unreachable, triggering an HA

Parameter	Description
	switchover to the redundant device. The parameter is measured in 10-msec units, meaning that whatever value you define for the parameter, the timeout is the value multiplied by 10. For example, if you configure the parameter to 100 (i.e., 100 x 10 = 1,000 msec = 1 second), an HA switchover happens after a 1-second loss of keep-alive messages. You can increase it, for example, to 300 (i.e., 300 x 10 = 3,000 msec = 3 seconds), which means that a switchover happens after a 3-second loss of keep-alive messages The valid value range is 100 to 6,000 (i.e., 1 minute). The default is 100. **Note:** For the parameter to take effect, a device reset is required.

HA Parameters

The High Availability (HA) parameters are described in the table below.

> When configuration is applied to the device whose MAC is the value of the HARemoteMAC parameter, all HA configuration is swapped between local and remote parameters, including the IP address of the Maintenance interface, which is swapped with the address configured for the HARemoteAddress parameter. For more information, see Quick-and-Easy Initial HA Configuration on page 834.

Table 60-21:HA Parameters

Parameter	Description
[HAMaintenanceIFDiffServValue]	Defines the DiffServ value for HA Maintenance traffic flowing on the HA Maintenance interface. The valid value is 0 (lowest priority or best-effort service) to 63 (highest priority). The default is 46. **Note:** For the parameter to take effect, a device reset is required.
`configure network > high-availability settings > operational-state-delay` [HAOperationalStateDelayInSec]	Defines the duration (in seconds) to delay the transition from HA non-operational state, which occurs during HA synchronization between active and redundant devices, to HA operational state. This feature may be useful, for example, to delay

Parameter	Description
	HA switchover when using switches with spanning tree protocol (STP) that take a long time until their ports (to which the redundant device is connected) is ready. In such a scenario, if this feature were not enabled (i.e., 0), after synchronization there would be no connectivity between the redundant device's network interface and the switch. The valid value is 0 to 180. The default is 0.
[HALocalMAC]	Specifies the MAC address of one of the two devices in the HA system. For more information, see Quick-and-Easy Initial HA Configuration on page 834. **Note:** When downloading an ini file from a device that is operating in HA mode, the parameter is the MAC address of the active device.
[HARemoteMAC]	Specifies the MAC address of one of the two devices in the HA system. For more information, see Quick-and-Easy Initial HA Configuration on page 834. **Note:** When downloading an ini file from a device that is operating in HA mode, the parameter is the MAC address of the redundant device.
'HA Device Name' `configure network > high-availability settings > unit-id-name` [HAUnitIdName]	Defines a name for the active device, which is displayed on the Home page to indicate the active device. The valid value is a string of up to 128 characters. The default value is "Device 1". **Note:** When the device sends alarms to OVOC, this name is displayed at the beginning of the alarm description in OVOC, for example, " (SBCSITE01): Proxy lost. looking for another proxy". However, the name is not displayed for the alarms retrieved (from the device's Active Alarms table) when OVOC **initially** connects to the device.
'Redundant HA Device Name' `configure network > high-`	Defines a name for the redundant device, which is displayed on the Home page to indicate the redundant device.

Parameter	Description
`availability settings > redundant-unit-id-name` [HARemoteUnitIdName]	The valid value is a string of up to 128 characters. The default value is "Device 2". **Note:** When the device sends alarms to OVOC, this name is displayed at the beginning of the alarm description in OVOC, for example, " (SBCSITE02): Proxy lost. looking for another proxy". However, the name is not displayed for the alarms retrieved (from the device's Active Alarms table) when OVOC **initially** connects to the device.
'HA Remote Address' `configure network > high-availability settings > remote-address` [HARemoteAddress]	Defines the Maintenance interface address of the redundant device in the HA system. By default, no value is defined. **Note:** For the parameter to take effect, a device reset is required.
'Preempt Mode' `configure network > high-availability settings > revertive-mode` [HARevertiveEnabled]	Enables HA switchover based on HA priority. ■ [0] **Disable** = (Default) A switchover over to the redundant device is done only if a failure occurs in the currently active device. ■ [1] **Enable** = A switchover over to the redundant device is done if a failure occurs in the currently active device. However, a switchover to the device with the highest priority (configured by the HAPriority parameter) occurs whenever the device recovers from a failure. Therefore, whenever possible, the highest priority device is the active one. For more information on the HA switchover mechanism, see Device Switchover upon Failure.
'Preempt Priority' `configure network > high-availability settings > priority` [HAPriority]	Defines the priority of the active device used in the HA Preempt mechanism. The valid value is 1 (lowest priority) to 10 (highest priority). The default is 5. **Note:** ■ The parameter is applicable only if you configure the 'Preempt Mode' parameter to Enable.

Parameter	Description
	■ You must configure each device in the HA system with different parameter values (priorities).
'Redundant Preempt Priority' `configure network > high-availability > redundant-priority` [HARemotePriority]	Defines the priority of the redundant device used in the HA Preempt mechanism. The valid value is 1 (lowest priority) to 10 (highest priority). The default is 5. **Note:** ■ The parameter is applicable only if you configure the 'Preempt Mode' parameter to Enable. ■ You must configure each device in the HA system with different parameter values (priorities).
HA Network Monitor Parameters For more information, see Monitoring IP Entities and HA Switchover upon Ping Failure on page 838.	
'HA Network Monitor' `configure network > high-availability settings > network-monitor-enabled` [HAPingEnabled]	Enables the HA Network Monitor feature, which checks connectivity with network destinations, using pings. When a user-defined number of destinations fail the connectivity test (see 'Network Monitor Threshold' parameter), an HA switchover occurs. ■ [0] **Disable** (default) ■ [1] **Enable**
'Monitor Threshold' `configure network > high-availability settings > network-monitor-threshold` [HaNetworkMonitorThreshold]	Defines the minimum number of monitored rows (configured in the HA Network Monitor table) whose destinations are unreachable that are required to trigger an HA switchover. The valid value is 1 to 10. The default is 1.

Security Parameters

This subsection describes the device's security parameters.

General Security Parameters

The general security parameters are described in the table below.

Table 60-22:General Security Parameters

Parameter	Description
'DNS Rebinding Protection' `configure system > web > dns-rebinding-protection-enabled` [DNSRebindingProtectionEnabled]	Enables protection against DNS rebinding attacks. This may occur when management users access the device using its hostname, configured by the [HostName] parameter, instead of the IP address. ■ [0] **Disable** (default) ■ [1] **Enable** For more information, see Enabling DNS Rebinding Protection on page 66
Media Latching	
'Inbound Media Latch Mode' `configure voip > media settings > inbound-media-latch-mode` [InboundMediaLatchMode]	Enables the Media Latching feature. ■ [0] **Strict** = The device is ready to receive (latch on to) media packets, but only if they are from a specific source IP address and UDP port, according to the remote IP address and UDP port in the negotiated SDP of the SIP message. **Note:** If the user agent is behind NAT and you have configured the [NATMode] parameter to [4] (**NAT By Signaling Restricted IP**), even if you have configured the 'Inbound Media Latch Mode' parameter to **Strict**, the device automatically changes it to **Dynamic**. ■ [1] **Dynamic** = (Default) Device latches on to the first stream. If it receives at least a minimum number of consecutive packets (configured by New<media type>StreamPackets) from a different source(s) and the device has not received packets from the current stream for a user-defined period (TimeoutToRelatch<media type>Msec), it latches on to the next packet received from any other stream. If other packets of a different media type are received from the new stream, based on IP address and SSRC for RTCP/RTP and based on IP address only for T.38, the packet is accepted immediately. **Note:** If a packet from the original (first latched onto) IP address:port is received at any

Parameter	Description
	time, the device latches on to this stream. ■ [2] **Dynamic-Strict** = Device latches on to the first stream. If it receives at least a minimum number of consecutive packets (configured by New<media type>StreamPackets) all from the same source which is different to the first stream and the device has not received packets from the current stream for a user-defined period (TimeoutToRelatch<media type>Msec), it latches on to the next packet received from any other stream. If other packets of different media type are received from the new stream based on IP address and SSRC for RTCP and based on IP address only for T.38, the packet is accepted immediately. **Note:** If a packet from the original (first latched onto) IP address:port is received at any time, the device latches on to this stream. ■ [3] **Strict-On-First** = Typically used for NAT, where the correct IP address:port is initially unknown. The device latches on to the stream received in the first packet. The device does not change this stream unless a packet is later received from the original source. **Note:** If you configure the parameter to [0] **Strict**, the device cannot perform NAT traversal. In this setup, configure the [NATMode] parameter to [1].
'New RTP Stream Packets' [NewRtpStreamPackets]	Defines the minimum number of continuous RTP packets received by the device's channel to allow latching onto the new incoming stream. The valid range is 0 to 20. The default is 3. If set to 0, the device is left exposed to attacks against multiple packet streams.
'New RTCP Stream Packets' [NewRtcpStreamPackets]	Defines the minimum number of continuous RTCP packets received by the device's channel to allow latching onto the new incoming stream. The valid range is 0 to 20. The default is 3. If set to 0, the device is left exposed to attacks against multiple packet streams.

Parameter	Description
'New SRTP Stream Packets' [NewSRTPStreamPackets]	Defines the minimum number of continuous SRTP packets received by the device's channel to allow latching onto the new incoming stream. The valid range is 0 to 20. The default is 3. If set to 0, the device is left exposed to attacks against multiple packet streams.
'New SRTCP Stream Packets' [NewSRTCPStreamPackets]	Defines the minimum number of continuous SRTCP packets received by the device's channel to allow latching onto the new incoming stream. The valid range is 0 to 20. The default is 3. If set to 0, the device is left exposed to attacks against multiple packet streams.
'Timeout To Relatch RTP' [TimeoutToRelatchRTPMsec]	Defines a period (msec) during which if no packets are received from the current RTP session, the channel can re-latch onto another stream. The valid range is any value from 0. The default is 200.
'Timeout To Relatch SRTP' [TimeoutToRelatchSRTPMsec]	Defines a period (msec) during which if no packets are received from the current SRTP session, the channel can re-latch onto another stream. The valid range is any value from 0. The default is 200.
'Timeout To Relatch Silence' [TimeoutToRelatchSilenceMsec]	Defines a period (msec) during which if no packets are received from the current RTP/SRTP session and the channel is in silence mode, the channel can re-latch onto another stream. The valid range is any value from 0. The default is 200.
'Timeout To Relatch RTCP' [TimeoutToRelatchRTCPMsec]	Defines a period (msec) during which if no packets are received from the current RTCP session, the channel can re-latch onto another RTCP stream. The valid range is any value from 0. The default is 10,000.
'Fax Relay Rx/Tx Timeout' [FaxRelayTimeoutSec]	Defines a period (sec) during which if no T.38 packets are received or sent from the current T.38 fax relay session, the channel can re-latch onto another stream.

Parameter	Description
	The valid range is 0 to 255. The default is 10.

HTTPS Parameters

The Secure Hypertext Transport Protocol (HTTPS) parameters are described in the table below.

Table 60-23:HTTPS Parameters

Parameter	Description
'Secured Web Connection (HTTPS)' `configure system > web > secured-connection` [HTTPSOnly]	Defines the application protocol for accessing the device's Web- or REST-based management interface. ■ [0] **HTTP and HTTPS** (default) ■ [1] **HTTPs Only** = Unencrypted HTTP packets are blocked. **Note:** For the parameter to take effect, a device reset is required.
`configure system > web > https-port` [HTTPSPort]	Defines the local Secured HTTPS port of the device. The parameter allows secure remote device Web- or REST-based management from the LAN. To enable secure Web management from the LAN, configure the desired port. The valid range is 1 to 65535 (other restrictions may apply within this range). The default port is 443. **Note:** For the parameter to take effect, a device reset is required.
'Require Client Certificates for HTTPS connection' `configure system > web > req-client-cert` [HTTPSRequireClientCertificate]	Enables the requirement of client certificates for HTTPS connection. ■ [0] **Disable** = (Default) Client certificates are not required. ■ [1] **Enable** = Client certificates are required. The client certificate must be preloaded to the device and its matching private key must be installed on the managing PC. Time and date must be correctly set on the device for the client certificate to be verified. **Note:** ■ For the parameter to take effect, a device reset is

Parameter	Description
	required. ■ For a description on implementing client certificates, see TLS for Remote Device Management.

SRTP Parameters

The Secure Real-Time Transport Protocol (SRTP) parameters are described in the table below.

Table 60-24:SRTP Parameters

Parameter	Description
'Media Security' `configure voip > media security > media-security-enable` [EnableMediaSecurity]	Enables Secure Real-Time Transport Protocol (SRTP). ■ [0] **Disable** (default) ■ [1] **Enable** **Note:**
'Master Key Identifier (MKI) Size' `configure voip > media security > srtp-tx-packet-mki-size` [SRTPTxPacketMKISize]	Global parameter that defines the size (in bytes) of the Master Key Identifier (MKI) in SRTP Tx packets. You can also configure this feature per specific calls, using IP Profiles (IpProfile_MKISize). For a detailed description of the parameter and for configuring this feature in the IP Profiles table, see Configuring IP Profiles. **Note:** If this feature is configured for a specific IP Profile, the settings of this global parameter is ignored for calls associated with the IP Profile.
'Symmetric MKI Negotiation' `configure voip > media security > symmetric-mki` [EnableSymmetricMKI]	Global parameter that enables symmetric MKI negotiation. You can also configure this feature per specific calls, using IP Profiles (IpProfile_EnableSymmetricMKI). For a detailed description of the parameter and for configuring this feature in the IP Profiles table, see

Parameter	Description
	Configuring IP Profiles. **Note:** If this feature is configured for a specific IP Profile, the settings of this global parameter is ignored for calls associated with the IP Profile.
'Offered SRTP Cipher Suites' `configure voip > media security > offer-srtp-cipher` [SRTPofferedSuites]	Defines the offered crypto suites (cipher encryption algorithms) for SRTP. ■ [0] **All** = (Default) All available crypto suites. ■ [1] **AES-CM-128-HMAC-SHA1-80** = device uses AES-CM encryption with a 128-bit key and HMAC-SHA1 message authentication with a 80-bit tag. ■ [2] **AES-CM-128-HMAC-SHA1-32** = device uses AES-CM encryption with a 128-bit key and HMAC-SHA1 message authentication with a 32-bit tag. ■ [16] **AES-256-CM-HMAC-SHA1-32** = AES-CM encryption with a 256-bit key and HMAC-SHA1 message authentication with a 32-bit tag. ■ [32] **AES-256-CM-HMAC-SHA1-80** = AES-CM encryption with a 256-bit key and HMAC-SHA1 message authentication with an 80-bit tag. **Note:** ■ The parameter also affects the selection of the crypto in the device's answer. For example, if the device receives an offer with two crypto lines ('a=crypto:') containing HMAC_SHA1_80 and HMAC_SHA_32, it uses the HMAC_SHA_32 key in its SIP 200

Parameter	Description
	OK response if the parameter is configured to **AES-CM-128-HMAC-SHA1-32**.
`configure voip > sbc settings > sbc-dtls-mtu` [SbcDtlsMtu]	Defines the maximum transmission unit (MTU) size for the DTLS handshake. The device does not attempt to send handshake packets that are larger than the configured value. Adjusting the MTU is useful when there are network constraints on the size of packets that can be sent. The valid value range is 228 to 1500. The default is 1400. **Note:** The parameter is applicable only to the SBC application.
`configure voip > sbc settings > dtls-time-between-transmissions` [DTLSTimeBetweenTransmissions]	Defines the minimum interval (in msec) that the device waits between transmission of DTLS packets in the same DTLS handshake. The configured value is applied in a "best-effort" manner (i.e., time between transmitted DTLS packets in the same handshake may differ due to constraints on the network layer and load on the device). The valid value is 0 (no forced delay between DTLS packet transmissions) to 100. The default is 5.
'Authentication on Transmitted RTP Packets' `configure voip > media security > RTP-authentication-disable-tx` [RTPAuthenticationDisableTx]	Enables authentication on transmitted RTP packets in a secured RTP session. ■ [0] **Enable** (default) ■ [1] **Disable**
'Encryption on Transmitted RTP Packets' `configure voip > media security > RTP-encryption-disable-tx` [RTPEncryptionDisableTx]	Enables encryption on transmitted RTP packets in a secured RTP session. ■ [0] **Enable** (default)

Parameter	Description
	■ [1] **Disable**
'Encryption on Transmitted RTCP Packets' `configure voip > media security > RTCP-encryption-disable-tx` [RTCPEncryptionDisableTx]	Enables encryption on transmitted RTCP packets in a secured RTP session. ■ [0] **Enable** (default) ■ [1] **Disable**
'SRTP Tunneling Authentication for RTP' `configure voip > media security > srtp-tnl-vld-rtp-auth` [SRTPTunnelingValidateRTPRxAuthentication]	Enables validation of SRTP tunneling authentication for RTP. ■ [0] **Disable** = (Default) The device does not perform any validation and forwards the packets as is. ■ [1] **Enable** = The device validates the packets (e.g., sequence number) and if successful, forwards the packets. If validation fails, it drops the packets. **Note:** ■ The parameter is applicable only to SRTP-to-SRTP calls and when both endpoints use the same authentication keys.
'SRTP Tunneling Authentication for RTCP' `configure voip > media security > srtp-tnl-vld-rtcp-auth` [SRTPTunnelingValidateRTCPRxAuthentication]	Enables validation of RTP tunneling authentication for RTCP. ■ [0] **Disable** = (Default) The device does not perform any validation and forwards the packets as is. ■ [1] **Enable** = The device validates the packets (e.g., sequence number) and if successful, forwards the packets. If validation fails, it drops the packets. **Note:** ■ The parameter is applicable only to SRTP-to-SRTP calls and when

Parameter	Description
	both endpoints use the same authentication keys.
`configure voip > sip-definition settings > srtp-state-behavior-mode` [ResetSRTPStateUponRekey]	Global parameter that enables synchronization of the SRTP state between the device and a server when a new SRTP key is generated upon a SIP session expire. You can also configure this feature per specific calls, using IP Profiles (IpProfile_ ResetSRTPStateUponRekey). For a detailed description of the parameter and for configuring this feature in the IP Profiles table, see Configuring IP Profiles. **Note:** If this feature is configured for a specific IP Profile, the settings of this global parameter is ignored for calls associated with the IP Profile.

TLS Parameters

The Transport Layer Security (TLS) parameters are described in the table below.

Table 60-25:TLS Parameters

Parameter	Description
'TLS Client Re-Handshake Interval' `configure network > security-settings > tls-re-hndshk-int` [TLSReHandshakeInterval]	Defines the time interval (in minutes) between TLS Re-Handshakes initiated by the device. The interval range is 0 to 1,500 minutes. The default is 0 (i.e., no TLS Re-Handshake).
'TLS Mutual Authentication' `configure network > security-settings > SIPSREQUIRECLIENTCERTIFICATE` [SIPSRequireClientCertificate]	Defines the device's mode of operation regarding mutual authentication and certificate verification for TLS connections. ■ [0] **Disable** = (Default) ✓ Device acts as a client: Verification of the server's certificate depends on the VerifyServerCertificate parameter.

Parameter	Description
	✓ Device acts as a server: The device does not request the client certificate. ■ [1] **Enable** = ✓ Device acts as a client: Verification of the server certificate is required to establish the TLS connection. ✓ Device acts as a server: The device requires the receipt and verification of the client certificate to establish the TLS connection. **Note:** ■ This feature can be configured per SIP Interface (see Configuring SIP Interfaces). ■ The SIPS certificate files can be changed using the parameters HTTPSCertFileName and HTTPSRootFileName.
'Peer Host Name Verification Mode' `configure network > security-settings > PEERHOSTNAMEVERIFICATIONMODE` [PeerHostNameVerificationMode]	Enables the device to verify the Subject Name of a TLS certificate received from SIP entities for authentication and establishing TLS connections. ■ [0] **Disable** (default) ■ [1] **Server Only** = Verify Subject Name only when acting as a client for the TLS connection. ■ [2] **Server & Client** = Verify Subject Name when acting as a server or client for the TLS connection. If the device receives a certificate from a SIP entity (IP Group) and the parameter is configured to **Server Only** or **Server & Client**, it attempts to authenticate the certificate based on the certificate's address. The device searches for a Proxy Set that

Parameter	Description
	contains the same address (IP address or FQDN) as that specified in the certificate's SubjectAltName (Subject Alternative Names). For Proxy Sets with an FQDN, the device checks the FQDN itself and **not** the DNS-resolved IP addresses. If a Proxy Set is found with a matching address, the device establishes a TLS connection. If a matching Proxy Set is not found, one of the following occurs: ■ If the certificate's SubjectAltName is marked as "critical", the device rejects the call. ■ If the SubjectAltName is not marked as "critical", the device checks if the FQDN in the certificate's Common Name (CN) of the SubjectName is the same as that configured for the TLSRemoteSubjectName parameter or for the Proxy Set. If they are the same, the device establishes a TLS connection; otherwise, the device rejects the call. **Note:** ■ If you configure the parameter to **Server & Client**, you also need to configure the SIPSRequireClientCertificate parameter to **Enable**. ■ For FQDN, the certificate may use wildcards (*) to replace parts of the domain name.
'TLS Client Verify Server Certificate' `configure network > security-settings > tls-vrfy-srvr-cert` [VerifyServerCertificate]	Determines whether the device, when acting as a client for TLS connections, verifies the Server certificate. The certificate is verified with the Root CA information. ■ [0] **Disable** (default) ■ [1] **Enable** **Note:** If Subject Name verification is

Parameter	Description
	necessary, the parameter PeerHostNameVerificationMode must be used as well.
'TLS Remote Subject Name' `configure network > security-settings > tls-rmt-subs-name` [TLSRemoteSubjectName]	Defines the Subject Name that is compared with the name defined in the remote side certificate when establishing TLS connections. If the SubjectAltName of the received certificate is not equal to any of the defined Proxies Host names/IP addresses and is not marked as 'critical', the Common Name (CN) of the Subject field is compared with this value. If not equal, the TLS connection is not established. If the CN uses a domain name, the certificate can also use wildcards ('*') to replace parts of the domain name. The valid range is a string of up to 49 characters. **Note:** The parameter is applicable only if the parameter PeerHostNameVerificationMode is set to 1 or 2.
'TLS Expiry Check Start' `expiry-check-start` [TLSExpiryCheckStart]	Defines when the device sends an SNMP alarm (acCertificateExpiryAlarm) to notify that the installed TLS server certificate is about to expire. This is defined by the number of days before the certificate's expiration date. For example, if configured to 5, the alarm is sent 5 days before the expiration date. For more information on the alarm, refer to the *SNMP Reference Guide*. The valid value is 0 to 3650. The default is 60.
'TLS Expiry Check Period' `expiry-check-period` [TLSExpiryCheckPeriod]	Defines the periodical interval (in days) for checking the TLS server certificate expiry date. The valid value is 1 to 3650. The default is 7.

SSH Parameters

Secure Shell (SSH) parameters are described in the table below.

Table 60-26:SSH Parameters

Parameter	Description
'Enable SSH Server' `configure system > cli-settings > ssh` [SSHServerEnable]	Enables the device's embedded SSH server. ■ [0] **Disable** ■ [1] **Enable** (default)
'Server Port' `configure system > cli-settings > ssh-port` [SSHServerPort]	Defines the port number for the embedded SSH server. Range is any valid port number. The default port is 22.
'Redundant Proxy Port' `configure system > cli-settings > ssh-red-device-port` [SSHRedundantProxyPort]	Defines the proxy SSH port number on the active device for accessing the redundant device's embedded SSH server from the active device for downloading files from the redundant device. The valid value is any valid port number. The default port is 0 (i.e., disabled). **Note:** ■ The parameter is applicable only when the device is in HA mode. ■ The port number must be different to the regular SSH port number, which is configured by the SSHServerPort parameter.
'Public Key' `configure system > cli-settings > ssh-require-public-key` [SSHRequirePublicKey]	Enables RSA public keys for SSH. ■ [0] **Disable** = (Default) RSA public keys are optional if a public key is configured. ■ [1] **Enable** = RSA public keys are mandatory. **Note:** ■ Public keys are configured per management user in the Local Users table (see Configuring Management User Accounts on page 48). ■ To define the key size, use the TLSPkeySize parameter.

Parameter	Description
'Max Payload Size' `ssh-max-payload-size` [SSHMaxPayloadSize]	Defines the maximum uncompressed payload size (in bytes) for SSH packets. The valid value is 550 to 32768. The default is 32768.
'Max Binary Packet Size' `configure system > cli-settings > ssh-max-binary-packet-size` [SSHMaxBinaryPacketSize]	Defines the maximum packet size (in bytes) for SSH packets. The valid value is 582 to 35000. The default is 35000.
'Maximum SSH Sessions' `configure system > cli-settings > ssh-max-sessions` [SSHMaxSessions]	Defines the maximum number of simultaneous SSH sessions. The valid range is 1 to 5. The default 5.
'Enable Last Login Message' `configure system > cli-settings > ssh-last-login-message` [SSHEnableLastLoginMessage]	Enables message display in SSH sessions of the time and date of the last SSH login. The SSH login message displays the number of unsuccessful login attempts since the last successful login. ■ [0] **Disable** ■ [1] **Enable** (default) **Note:** The last SSH login information is cleared when the device is reset.
'Max Login Attempts `configure system > cli-settings > ssh-max-login-attempts` [SSHMaxLoginAttempts]	Defines the maximum SSH login attempts allowed for entering an incorrect password by an administrator before the SSH session is rejected. The valid range is 1 to 5. The default is 3. **Note:** The new setting takes effect only for new subsequent SSH connections.

IDS Parameters

The Intrusion Detection System (IDS) parameters are described in the table below.

Table 60-27:IDS Parameters

Parameter	Description
'Intrusion Detection System	Enables the IDS feature.

Parameter	Description
(IDS)' enable-ids [EnableIDS]	■ [0] **Disable** (default) ■ [1] **Enable**
'Alarm Clear Period' alarm-clear-period [IDSAlarmClearPeriod]	Defines the interval (in seconds) after which an IDS alarm is cleared from the Active Alarms table if no thresholds are crossed during this time. However, this "quiet" period must be at least twice the Threshold Window value. For example, if IDSAlarmClearPeriod is set to 20 sec and the Threshold Window is set to 15 sec, the IDSAlarmClearPeriod parameter is ignored and the alarm is cleared only after 30 seconds (2 x 15 sec). The valid value is 0 to 86400. The default is 300.
'Excluded Response Codes' excluded-responses [IDSExcludedResponseCodes]	Defines the SIP response codes that are excluded form the IDS count for SIP dialog establishment failures. The valid value is 400 through to 699. The maximum length is 100 characters. You can configure the parameter with multiple codes, where each code is separated by a comma (without spaces). The default is 408,422,423,480,481,486,487,500,501,502,503,504,505,600. For more information, see Configuring SIP Response Codes to Exclude from IDS on page 199. **Note:** ■ The parameter applies only to rejected responses received from the remote network; not rejected responses generated by the device (except for 404). ■ The response codes 401 and 407 are considered authentication failures and therefore, are not applicable to this parameter.

OCSP Parameters

The Online Certificate Status Protocol (OCSP) parameters are described in the table below.

Table 60-28:OCSP Parameters

Parameter	Description
'Enable OCSP Server' configure network > ocsp > enable	Enables or disables certificate checking using OCSP.

Parameter	Description
[OCSPEnable]	■ [0] **Disable** (default) ■ [1] **Enable** For a description of OCSP, see Configuring Certificate Revocation Checking (OCSP).
'Primary Server IP' `configure network > ocsp > server-ip` [OCSPServerIP]	Defines the IP address of the OCSP server. The default IP address is 0.0.0.0.
'Secondary Server IP' `configure network > ocsp > secondary-server-ip` [OCSPSecondaryServerIP]	Defines the IP address (in dotted-decimal notation) of the secondary OCSP server (optional). The default IP address is 0.0.0.0.
'Server Port' `configure network > ocsp > server-port` [OCSPServerPort]	Defines the OCSP server's TCP port number. The default port number is 2560.
'Default Response When Server Unreachable' `configure network > ocsp > default-response` [OCSPDefaultResponse]	Determines whether the device allows or rejects peer certificates when the OCSP server cannot be contacted. ■ [0] **Reject** (default) ■ [1] **Allow**

Proxy, Registration and Authentication Parameters

The proxy server, registration and authentication SIP parameters are described in the table below.

Table 60-29:Proxy, Registration and Authentication SIP Parameters

Parameter	Description
'Proxy Name' `configure voip > sip-definition proxy-and-registration > proxy-name`	Defines the Home Proxy domain name. If specified, this name is used as the Request-URI in REGISTER, INVITE and other SIP messages, and as the host part of the To header in INVITE messages. If not

Parameter	Description
[ProxyName]	specified, the Proxy IP address is used instead. The valid value is a string of up to 49 characters. **Note:** The parameter functions together with the UseProxyIPasHost parameter.
'Use Proxy IP as Host' `configure voip > sip-definition proxy-and-registration > use-proxy-ip-as-host` [UseProxyIPasHost]	Enables the use of the proxy server's IP address (in dotted-decimal notation) as the host name in SIP From and To headers in REGISTER requests. ■ [0] **Disable** (default) ■ [1] **Enable** If the parameter is disabled and the device registers to an IP Group (i.e., proxy server), it uses the string configured by the ProxyName parameter as the host name in the REGISTER's Request-URI and uses the string configured by the IP Groups table parameter, SIPGroupName as the host name in the To and From headers. If the IP Group is configured with a Proxy Set that has multiple IP addresses, all the REGISTER messages sent to these proxies are sent with the same host name. **Note:** If the parameter is disabled and the ProxyName parameter is not configured, the proxy's IP address is used as the host name in the REGISTER Request-URI.
'Redundancy Mode' `configure voip > sip-definition proxy-and-registration > redundancy-mode` [ProxyRedundancyMode]	Determines whether the device switches back to the primary Proxy after using a redundant Proxy. ■ [0] **Parking** = (Default) The device continues working with a redundant (now active) Proxy until the next failure, after which it works with the next redundant Proxy. ■ [1] **Homing** = The device always tries to work with the primary Proxy server (i.e., switches back to the primary Proxy whenever it's available). **Note:** To use this Proxy Redundancy mechanism, you need to enable proxy keep-alive (see the [ProxySet_ProxyKeepAliveTime] parameter).
'Proxy IP List Refresh Time'	Defines the interval (in seconds) at which the

Parameter	Description
`configure voip > sip-definition proxy-and-registration > proxy-ip-lst-rfrsh-time` [ProxyIPListRefreshTime]	device performs DNS resolution for Proxy Sets that are configured with an FQDN (host name), in order to translate (resolve) it into IP addresses. The device maintains a cache of DNS resolutions, and uses the cached responses as long as the TTL has not expired. If the TTL has expired, a new DNS request is sent to the DNS server. For example, if configured to 60, the device queries the DNS server every 60 seconds. if successful, the device refreshes the Proxy Set's list of DNS-resolved IP addresses. The device caches (stores) the DNS-resolved IP addresses of the last successful DNS query. It clears every entry in the cache 30 minutes after its time-to-live (TTL) value expires. If the DNS server is offline (for whatever reason) when the device performs a DNS query, instead of taking the Proxy Set offline, the device reuses the cached DNS-resolved addresses. In such a scenario, the device continues to query the DNS server every 10 seconds. However, if the DNS server is still offline after the device has deleted the cached DNS-resolved IP addresses, the device takes the Proxy Set offline. The valid value is 0, or 5 to 2,000,000. The default is 60. The value 0 disables periodic DNS queries and DNS resolution is done only once - upon device reset, device power up, or new and modified configuration.
'Always Use Proxy' `configure voip > sip-definition proxy-and-registration > always-use-proxy` [AlwaysSendToProxy]	Determines whether the device sends SIP messages and responses through a Proxy server. ■ [0] **Disable** = (Default) Use standard SIP routing rules. ■ [1] **Enable** = All SIP messages and responses are sent to the Proxy server. **Note:** The parameter is applicable only if a Proxy server is used (i.e., the parameter IsProxyUsed is set to 1).
'DNS Query Type'	Enables the use of DNS Naming Authority Pointer

Parameter	Description
`configure voip > sip-definition proxy-and-registration > dns-query` [DNSQueryType]	(NAPTR) and Service Record (SRV) queries to resolve Proxy and Registrar servers and to resolve all domain names that appear in the SIP Contact and Record-Route headers. ■ [0] **A-Record** = (Default) No NAPTR or SRV queries are performed. ■ [1] **SRV** = If the Proxy/Registrar IP address parameter, Contact/Record-Route headers, or IP address configured in the routing tables contain a domain name, an SRV query is performed. The device uses the first host name received from the SRV query. The device then performs a DNS A-record query for the host name to locate an IP address. ■ [2] **NAPTR** = An NAPTR query is performed. If it is successful, an SRV query is sent according to the information received in the NAPTR response. If the NAPTR query fails, an SRV query is performed according to the configured transport type. **Note:** ■ If the Proxy/Registrar IP address parameter, the domain name in the Contact/Record-Route headers, or the IP address configured in the routing tables contain a domain name with a port definition, the device performs a regular DNS A-record query. ■ If a specific Transport Type is configured, a NAPTR query is not performed. ■ To enable NAPTR/SRV queries for Proxy servers only, use the global parameter ProxyDNSQueryType, or use the Proxy Sets table.
'Proxy DNS Query Type' `configure voip > sip-definition proxy-and-registration > proxy-dns-query` [ProxyDNSQueryType]	Global parameter that defines the DNS query record type for resolving the Proxy server's configured domain name (FQDN) into an IP address. ■ [0] **A-Record** = (Default) A-record DNS query.

Parameter	Description
	■ [1] **SRV** = If the Proxy IP address parameter contains a domain name without port definition (e.g., ProxyIP = domain.com), an SRV query is performed. The SRV query returns up to four Proxy host names and their weights. The device then performs DNS A-record queries for each Proxy host name (according to the received weights) to locate up to four Proxy IP addresses. Thus, if the first SRV query returns two domain names and the A-record queries return two IP addresses each, no additional searches are performed. ■ [2] **NAPTR** = NAPTR query is done. If successful, an SRV query is sent according to the information received in the NAPTR response. If the NAPTR query fails, an SRV query is done according to the configured transport type. If the Proxy IP address parameter contains a domain name with port definition (e.g., ProxyIP = domain.com:5080), the device performs a regular DNS A-record query. If a specific Transport Type is defined, a NAPTR query is not performed. **Note:** ■ This feature can be configured per Proxy Set in the Proxy Sets table (see Configuring Proxy Sets). ■ When enabled, NAPTR/SRV queries are used to discover Proxy servers even if the parameter DNSQueryType is disabled.
'Use Gateway Name for OPTIONS' `configure voip > sip-definition proxy-and-registration > use-gw-name-for-opt` [UseGatewayNameForOptions]	Defines if the device's IP address, proxy's IP address, or device's name is used as the host part for the Request-URI in keep-alive SIP OPTIONS messages sent to the proxy (if enabled). The device uses the OPTIONS messages as a keep-alive with its primary and redundant SIP proxy servers. Proxy keep-alive by SIP OPTIONS is enabled per Proxy Set in the Proxy Sets table, by configuring the [ProxySet_EnableProxyKeepAlive] parameter to [1]). For more information, see Configuring Proxy

Parameter	Description
	Sets. ■ [0] **No** = (Default) The device's IP address is used in the keep-alive OPTIONS messages. ■ [1] **Yes** = The device's name, configured by the [SIPGatewayName] parameter, is used in the keep-alive OPTIONS messages. ■ [2] **Server** = The proxy's IP address is used in the SIP From and To headers in the keep-alive OPTIONS messages.
[FailedOptionsRetryTime]	Defines how long the device waits (in seconds) before re-sending a SIP OPTIONS keep-alive message to the proxy once the device considers the proxy as offline (i.e., after all retransmissions, configured by the [ProxySet_ FailureDetectionRetransmissions] parameter, have failed). The valid value range is 1 to 3600. The default is 1. **Note:** ■ The parameter is applicable only if you enable proxy keep-alive by SIP OPTIONS messages (i.e., [ProxySet_EnableProxyKeepAlive] = [1]). ■ A failed SIP response can be no response or a response specified by the [ProxySet_ KeepAliveFailureResp] parameter.
`configure voip > sbc settings > abort-retries-on-icmp-error` [AbortRetriesOnICMPError]	When using UDP as the transport protocol, the retries failed transmissions to a proxy server is according to the [ProxySet_ FailureDetectionRetransmissions] parameter. However, when the failed attempt receives an ICMP error (which indicates Host Unreachable or Network Unreachable) as opposed to a timeout, it may be desirable to abandon additional retries in favor of trying the next IP address (proxy server) in the Proxy Set. This is often desirable when Proxy Hot Swap is enabled. ■ [0] = Disable. The retries the same proxy according to the [ProxySet_ FailureDetectionRetransmissions] parameter

Parameter	Description
	(regardless of error response type). ■ [1] = (Default) Enable. Upon the receipt of an ICMP error response, the doesn't try the proxy again (i.e., ignores the [ProxySet_ FailureDetectionRetransmissions] parameter), but instead tries the next proxy in the Proxy Set.
'Password' `configure voip > sip-definition proxy-and-registration > auth-password` [AuthPassword]	Defines the password for Basic/Digest authentication with a Proxy/Registrar server. A single password is used for all device ports. The default is 'Default_Passwd'. **Note:** ■ The parameter cannot be configured with wide characters.
'Cnonce' `configure voip > sip-definition proxy-and-registration > cnonce-4-auth` [Cnonce]	Defines the Cnonce string used by the SIP server and client to provide mutual authentication. The value is free format, i.e., 'Cnonce = 0a4f113b'. The default is 'Default_Cnonce'.
'Challenge Caching Mode' `configure voip > sip-definition proxy-and-registration > challenge-caching` [SIPChallengeCachingMode]	Enables local caching of SIP message authorization challenges from Proxy servers. The device sends the first request to the Proxy without authorization. The Proxy sends a 401/407 response with a challenge for credentials. The device saves (caches) the response for further uses. The device sends a new request with the appropriate credentials. Subsequent requests to the Proxy are automatically sent with credentials (calculated from the saved challenge). If the Proxy doesn't accept the new request and sends another challenge, the old challenge is replaced with the new one. One of the benefits of the feature is that it may reduce the number of SIP messages transmitted through the network. ■ [0] **None** = (Default) Challenges are not cached. Every new request is sent without preliminary authorization. If the request is challenged, a

Parameter	Description
	new request with authorization data is sent. ■ [1] **INVITE Only** = Challenges issued for INVITE requests are cached. This prevents a mixture of REGISTER and INVITE authorizations. ■ [2] **Full** = Caches all challenges from the proxies. **Note:** ■ Challenge caching is used with all proxies and not only with the active one. ■ The challenge can be cached per Account or per user whose credentials are configured in the User Information table.
Registrar Parameters	
'Registration Time' `configure voip > sip-definition proxy-and-registration > registration-time` [RegistrationTime]	Defines the time interval (in seconds) for registering to a Proxy server. The value is used in the SIP Expires header. The parameter also defines the time interval between Keep-Alive messages when the parameter EnableProxyKeepAlive is set to 2 (REGISTER). Typically, the device registers every 3,600 sec (i.e., one hour). The device resumes registration according to the parameter RegistrationTimeDivider. The valid range is 10 to 2,000,000. The default is 180.
'Re-registration Timing [%]' `configure voip > sip-definition proxy-and-registration > re-registration-timing` [RegistrationTimeDivider]	Defines the re-registration timing (in percentage). The timing is a percentage of the re-register timing set by the Registrar server. The valid range is 50 to 100. The default is 50. For example: If the parameter is set to 70% and the Registration Expires time is 3600, the device re-sends its registration request after 3600 x 70% (i.e., 2520 sec). **Note**: The parameter may be overridden if the parameter RegistrationTimeThreshold is greater than 0.
'Registration Retry Time'	Defines the time interval (in seconds) after which a

Parameter	Description
`configure voip > sip-definition proxy-and-registration > registration-retry-time` [RegistrationRetryTime]	registration request is re-sent if registration fails with a 4xx response or if there is no response from the Proxy/Registrar server. The default is 30 seconds. The range is 10 to 3600. **Note:** Registration retry time can also be configured with the MaxRegistrationBackoffTime parameter.
'Max Registration Backoff Time' `configure voip > sip-definition proxy-and-registration > max-registration-backoff-time` [MaxRegistrationBackoffTime]	Defines a dynamic time-to-wait interval before the device attempts to register the SIP entity again after a registration failure. The parameter is applicable only to registrations initiated by the device on behalf of SIP entities (for example, User Info, Accounts, Endpoints or the device itself) with a SIP proxy server (registrar). The valid value is 0 to 3000000 (i.e., 3 million seconds). The default is 0 (i.e., disabled). In contrast to the RegistrationRetryTime parameter, which defines a fixed time to wait between registration attempts due to registration failure, this parameter configures the device to increase the time-to-wait interval for each subsequent registration attempt (per RFC 5626, Section 4.5) for a specific registration flow. In other words, the interval changes between registration attempts. The parameter operates together with the RegistrationRetryTime parameter. When the MaxRegistrationBackoffTime parameter is configured, the wait-time before another registration attempt increases after each failed registration (until it reaches the maximum value specified by the parameter). The device uses the following algorithm to calculate the incremental augmented wait-time between each registration attempt: `Wait Time = min (max-time, (base-time * (2 ^ consecutive-failures)))` Where: ■ *max-time* is the value configured by

Parameter	Description
	MaxRegistrationBackoffTime ■ *base-time* is the value configured by RegistrationRetryTime For example, if *max-time* is 1800 seconds and *base-time* is 30 seconds, and there were three consecutive registration failures, then the upper-bound wait time is the minimum of (1800, 30* (2^3)), which is (1800, 240) and thus, the minimum of the two values is 240 (seconds). The actual time the device waits before retrying registration is computed by a uniform random time between 50% and 100% of the upper-bound wait time (e.g., for an upper-bound wait-time of 240, the actual wait-time is between 120 and 240 seconds). As can be seen from the algorithm, the upper-bound wait time can never exceed the value of the MaxRegistrationBackoffTime parameter.
'Registration Time Threshold' `configure voip > sip-definition proxy-and-registration > registration-time-thres` [RegistrationTimeThreshold]	Defines a threshold (in seconds) for re-registration timing. If the parameter is greater than 0, but lower than the computed re-registration timing (according to the parameter RegistrationTimeDivider), the re-registration timing is set to the following: timing set by the Registration server in the SIP Expires header minus the value of the parameter RegistrationTimeThreshold. The valid range is 0 to 2,000,000. The default is 0.
'ReRegister On Connection Failure' `configure voip > sip-definition proxy-and-registration > reg-on-conn-failure` [ReRegisterOnConnectionFailure]	Enables the device to perform SIP re-registration upon TCP/TLS connection failure. ■ [0] **Disable** (default) ■ [1] **Enable**
`configure voip > sip-definition proxy-and-registration > expl-un-reg` [UnregistrationMode]	Enables the device to perform explicit unregisters. ■ [0] **Disable** (default) ■ [1] **Enable** = The device sends an asterisk ("*") value in the SIP Contact header, instructing the

Parameter	Description
	Registrar server to remove all previous registration bindings. The device removes SIP User Agent (UA) registration bindings in a Registrar, according to RFC 3261. Registrations are soft state and expire unless refreshed, but they can also be explicitly removed. A client can attempt to influence the expiration interval selected by the Registrar. A UA requests the immediate removal of a binding by specifying an expiration interval of "0" for that contact address in a REGISTER request. UA's should support this mechanism so that bindings can be removed before their expiration interval has passed. Use of the "*" Contact header field value allows a registering UA to remove all bindings associated with an address-of-record (AOR) without knowing their precise values. **Note:** The REGISTER-specific Contact header field value of "*" applies to all registrations, but it can only be used if the Expires header field is present with a value of "0".
'Add Empty Authorization Header' `configure voip > sip-definition settings > add-empty-author-hdr` [EmptyAuthorizationHeader]	Enables the inclusion of the SIP Authorization header in initial registration (REGISTER) requests sent by the device. ■ [0] **Disable** (default) ■ [1] **Enable** The Authorization header carries the credentials of a user agent (UA) in a request to a server. The sent REGISTER message populates the Authorization header with the following parameters: ■ username - set to the value of the private user identity ■ realm - set to the domain name of the home network ■ uri - set to the SIP URI of the domain name of the home network ■ nonce - set to an empty value ■ response - set to an empty value

Parameter	Description
	For example: Authorization: Digest username=alice_ private@home1.net, realm="home1.net", non-ce="", response="e56131d19580cd833064787ecc" **Note:** This registration header is according to the IMS 3GPP TS24.229 and PKT-SP-24.220 specifications.

'Add initial Route Header'
`configure voip > sip-definition proxy-and-registration > add-init-rte-hdr`
[InitialRouteHeader]
	Enables the inclusion of the SIP Route header in initial registration or re-registration (REGISTER) requests sent by the device.
	■ [0] **Disable** (default)
	■ [1] **Enable**
	When the device sends a REGISTER message, the Route header includes either the Proxy's FQDN, or IP address and port according to the configured Proxy Set, for example:
	`Route: <sip:10.10.10.10;lr;transport=udp>`
	or
	`Route: <sip: pcscf-gm.ims.rr.com;lr;transport=udp>`

`configure voip > sip-definition proxy-and-registration > ping-pong-keep-alive` [UsePingPongKeepAlive]	Enables the use of the carriage-return and line-feed sequences (CRLF) Keep-Alive mechanism, according to RFC 5626 "Managing Client-Initiated Connections in the Session Initiation Protocol (SIP)" for reliable, connection-orientated transport types such as TCP. ■ [0] **Disable** (default) ■ [1] **Enable** The SIP user agent/client (i.e., device) uses a simple periodic message as a keep-alive mechanism to keep their flow to the proxy or registrar alive (used for example, to keep NAT bindings open). For connection-oriented transports such as TCP/TLS this is based on CRLF. This mechanism uses a client-to-server "ping" keep-alive and a corresponding server-to-client "pong"

Parameter	Description
	message. This ping-pong sequence allows the client, and optionally the server, to tell if its flow is still active and useful for SIP traffic. If the client does not receive a pong in response to its ping, it declares the flow “dead” and opens a new flow in its place. In the CRLF Keep-Alive mechanism the client periodically (defined by the PingPongKeepAliveTime parameter) sends a double-CRLF (the "ping") then waits to receive a single CRLF (the "pong"). If the client does not receive a "pong" within an appropriate amount of time, it considers the flow failed. **Note:** The device sends a CRLF message to the Proxy Set only if the Proxy Keep-Alive feature (EnableProxyKeepAlive parameter) is enabled and its transport type is set to TCP or TLS. The device first sends a SIP OPTION message to establish the TCP/TLS connection and if it receives any SIP response, it continues sending the CRLF keep-alive sequences.
`configure voip > sip-definition proxy-and-registration > ping-pong-keep-alive-time` [PingPongKeepAliveTime]	Defines the periodic interval (in seconds) after which a “ping” (double-CRLF) keep-alive is sent to a proxy/registrar, using the CRLF Keep-Alive mechanism. The default range is 5 to 2,000,000. The default is 120. The device uses the range of 80-100% of this user-defined value as the actual interval. For example, if the parameter value is set to 200 sec, the interval used is any random time between 160 to 200 seconds. This prevents an “avalanche” of keep-alive by multiple SIP UAs to a specific server.
'Max Generated Register Rate' `configure voip > sip-definition proxy-and-registration > max-gen-reg-rate` [MaxGeneratedRegistersRate]	Defines the maximum number of user register requests (REGISTER messages) that the device sends (to a proxy or registrar server) at a user-defined rate configured by the GeneratedRegistersInterval parameter. The parameter is useful in that it may be used to prevent an overload on the device's CPU caused by sending many registration requests at a given time. The valid value is 30 to 300 register requests per

Parameter	Description
	second. The default is 150. For configuration examples, see the description of the GeneratedRegistersInterval parameter.
'Generated Register Interval' `configure voip > sip-definition proxy-and-registration sip-definition settings > gen-reg-int` [GeneratedRegistersInterval]	Defines the rate (in seconds) at which the device sends user register requests (REGISTER messages). The parameter is based on the maximum number of REGISTER messages that can be sent at this rate, configured by the MaxGeneratedRegistersRate parameter. The valid value is 1 to 5. The default is 1. Configuration examples: ■ If you configure the MaxGeneratedRegistersRate parameter to 100 and the GeneratedRegistersInterval to 5, the device sends a maximum of 20 REGISTER messages per second (i.e., 100 messages divided by 5 sec; 100 per 5 seconds). ■ If you configure the MaxGeneratedRegistersRate parameter to 100 and the GeneratedRegistersInterval to 1, the device sends a maximum of a 100 REGISTER messages per second.
`configure voip > sip-definition settings > account-invite-failure-trigger-codes` [AccountInviteFailureTriggerCodes]	Defines SIP response codes that if received for a failed INVITE message sent for an Account, triggers the device to re-register the Account. The parameter is applicable only if the Account's 'Re-Register on Invite Failure' parameter in the Accounts table is configured to **Enable** (see Configuring Registration Accounts on page 573). The valid value is a SIP response code. Multiple response codes can be configured, where each value is separated by a comma. The default is "403,408,480" (without quotation marks). **Note:** SIP response code 408 also refers to an INVITE timeout (i.e., no reply from server). Therefore, if re-registration is needed for such a scenario, make sure that you configure the parameter with "408" as well.
`configure voip > sip-`	Enables the device to retry registering even if the

Parameter	Description
`definition settings > ignore-auth-stale` [IgnoreAuthorizationStale]	last SIP 401\407 response included "stale=false". When the device initiates a REGISTER request with an Authorization header (according to the relevant configured credentials), and it receives a SIP 401\407 response with the stale parameter set to "false", by default the device doesn't try to send another REGISTER message. When the parameter is enabled, the device retries registering even if the last 401\407 response had "stale=false". ■ [0] = (Default) If the device receives a SIP 401\407 response with "stale=true" or no stale parameter at all, it sends another REGISTER message. If "stale=false", the device doesn't send another REGISTER message. ■ [1] = If the device receives a SIP 401\407 response with "stale=false", it sends another REGISTER message. **Note:** This parameter is applicable only to REGISTER requests which the device initiates (e.g., for an Account or for Gateway endpoints); it's not for REGISTER requests that the device forwards from the user to the registrar server.
`configure voip >sip-definition settings > authenticated-message-handling` [AuthenticatedMessageHandling]	Defines if a Message Manipulation Set is run again on incoming authenticated SIP messages received after the device sends a SIP 401 response for challenging initial incoming SIP REGISTER requests. Typically, this is not required and the rules of a Message Manipulation Set that are configured to run on incoming REGISTER requests are applied only when the initial unauthenticated REGISTER request is received. However, if the Message Manipulation Set includes a Message Manipulation rule that specifies that manipulation must be done on the SIP Authorization header (i.e., 'Condition' parameter value is "Header.Authorization !exists"), which is present only in authenticated messages, then configure the parameter to [1]. ■ [0] = (Default) Disable - The Message Manipulation Set is not run again on authenticated messages and only applied to

Parameter	Description
	initial unauthenticated messages. The device uses this manipulated initial REGISTER request for further processing (e.g., classification or routing). ■ [1] = The Message Manipulation Set is run again on authenticated messages (if it includes a rule whose condition is the Authorization header). The device uses this manipulated authenticated REGISTER request for further processing (e.g., classification or routing).

Network Application Parameters

The SIP network application parameters are described in the table below.

Table 60-30:SIP Network Application Parameters

Parameter	Description
`configure voip > sip-definition settings > tcp-keepalive-time` [TCPKeepAliveTime]	Defines the interval (in sec) between the last data packet sent and the first keep-alive probe to send. The valid value is 10 to 65,000. The default is 60. **Note:** ■ Simple ACKs such as keepalives are not considered data packets. ■ TCP keepalive is enabled per SIP Interface in the SIP Interfaces table.
`configure voip > sip-definition settings > tcp-keepalive-interval` [TCPKeepAliveInterval]	Defines the interval (in sec) between consecutive keep-alive probes, regardless of what the connection has exchanged in the meantime. The valid value is 10 to

Parameter	Description
	65,000. The default is 10. **Note:** TCP keepalive is enabled per SIP Interface in the SIP Interfaces table.
`configure voip > sip-definition settings > tcp-keepalive-retry` [TCPKeepAliveRetry]	Defines the number of unacknowledged keep-alive probes to send before considering the connection down. The valid value is 1 to 100. The default is 5. **Note:** TCP keepalive is enabled per SIP Interface in the SIP Interfaces table.

General SIP Parameters

The general SIP parameters are described in the table below.

Table 60-31:General SIP Parameters

Parameter	Description
`configure voip > sip-definition settings > max-sdp-sess-ver-id` [MaxSDPSessionVersionId]	Defines the maximum number of characters allowed in the SDP body's "o=" (originator and session identifier) field for the session ID and session version values. An example of an "o=" line with session ID and session version values (in bold) is shown below: `o=jdoe **2890844526 2890842807** IN IP4 10.47.16.5` The valid value range is 1,000 to 214,748,3647 (default).
[UseRandomUser]	Enables the device to generate a random string value for the user part of the SIP Contact header in the REGISTER message for the registration of user Accounts with the device. To configure Accounts, see Configuring Registration Accounts. The string includes letters and may include numbers, but it always begins with a letter. The string is unique to each Account. An example of a randomly assigned user part is shown (in bold) below:

Parameter	Description
	`Contact: <sip:`**`HRaNEmZnfX6xZl4`**`@pc33.atlanta.com>` ■ [0] = (Default) Disable ■ [1] = Enable. The device generates **one** unique string for the user part per Account. Each Account registers with its unique user part string. All INVITE messages for this new Account are sent with this unique user part. This same unique user part string is also used for registration refreshes and for un-registering the Account. The device stops using the random user part in the following scenarios: ■ The user sends an unregister request. ■ The device sends a REGISTER request for the user, but does not receive a SIP 200 OK in response. ■ The parameter is disabled. When enabled again, new random user parts are assigned to the Accounts.
`configure voip > sip-definition settings > unreg-on-startup` [UnregisterOnStartup]	Enables the device to unregister all user Accounts that were registered with the device, upon a device reset. During device start-up, each Account sends a REGISTER message (containing "Contact: *") to unregister all contact URIs belonging to its Address-of-Record (AOR), and then a second after they are unregistered, the device re-registers the Account. ■ [0] = (Default) Disable ■ [1] = Enable To configure Accounts, see Configuring Registration Accounts.
`configure voip > sip-definition settings > sync-ims-accounts` [SyncIMSAccounts]	Enables synchronization of multiple Accounts per the IMS specification. ■ [0] = (Default) Disable ■ [1] = Enable To configure Accounts, see Configuring Registration Accounts. For more information on multiple Accounts

Parameter	Description
	synchronization per IMS, see Synchronizing Multiple SIP Accounts per IMS Specification on page 584.
'Send Reject (503) upon Overload' `configure voip > sip-definition settings > reject-on-ovrld` [SendRejectOnOverload]	Disables the sending of SIP 503 (Service Unavailable) responses upon receipt of new SIP dialog-initiating requests when the device's CPU is overloaded and thus, unable to accept and process new SIP messages. ■ [0] **Disable** = No SIP 503 response is sent when CPU overloaded. ■ [1] **Enable** = (Default) SIP 503 response is sent when CPU overloaded. **Note:** Even if the parameter is disabled (i.e., 503 is not sent), the device still discards the new SIP dialog-initiating requests when the CPU is overloaded.
'SIP 408 Response upon non-INVITE' `configure voip > sip-definition settings > enbl-non-inv-408` [EnableNonInvite408Reply]	Enables the device to send SIP 408 responses (Request Timeout) upon receipt of non-INVITE transactions. Disabling this response complies with RFC 4320/4321. By default, and in certain circumstances such as a timeout expiry, the device sends a SIP 408 Request Timeout in response to non-INVITE requests (e.g., REGISTER). ■ [0] **Disable** = SIP 408 response is not sent upon receipt of non-INVITE messages (to comply with RFC 4320). ■ [1] **Enable** = (Default) SIP 408 response is sent upon receipt of non-INVITE messages, if necessary.
'Remote Management by SIP NOTIFY' configure voip > sip-definition settings > sip-remote-reset [EnableSIPRemoteReset]	Enables a specific device action upon the receipt of a SIP NOTIFY request, where the action depends on the value in the Event header. ■ [0] **Disable** (default) ■ [1] **Enable** The action depends on the Event header value: ■ "Event: check-sync;reboot=false": Triggers the regular Automatic Update feature (if Automatic Update has been enabled on the device). ■ "Event: check-sync;reboot=true": Triggers a device reset.

Parameter	Description
	■ "Event: soft-sync": Triggers the device to disconnect all current calls. ■ If the 'reboot=' parameter is not specified in the Event header, it defaults to 'true' (i.e., triggers a restart). **Note:** The Event header value is proprietary to AudioCodes.
'Max SIP Message Length' [MaxSIPMessageLength]	Defines the maximum size (in Kbytes) for each SIP message that can be sent over the network. The device rejects messages exceeding this user-defined size. The valid value range is 1 to 100. The default is 100.
[SIPForceRport]	Determines whether the device sends SIP responses to the UDP port from where SIP requests are received even if the 'rport' parameter is not present in the SIP Via header. ■ [0] = (Default) Disabled. The device sends the SIP response to the UDP port defined in the Via header. If the Via header contains the 'rport' parameter, the response is sent to the UDP port from where the SIP request is received. ■ [1] = Enabled. SIP responses are sent to the UDP port from where SIP requests are received even if the 'rport' parameter is not present in the Via header.
'Reject Cancel after Connect' `configure voip > sip-definition settings > reject-cancel-after-connect` [RejectCancelAfterConnect]	Enables or disables the device to accept or reject SIP CANCEL requests received after the receipt of a 200 OK in response to an INVITE (i.e., call established). According to the SIP standard, a CANCEL can be sent only during the INVITE transaction (before 200 OK), and once a 200 OK response is received the call can be rejected only by a BYE request. ■ [0] **Disable** = (Default) Accepts a CANCEL request received during the INVITE transaction by sending a 200 OK response and terminates the call session. ■ [1] **Enable** = Rejects a CANCEL request received during the INVITE transaction by sending a SIP 481 (Call/Transaction Does Not Exist) response and maintains the call session.

Parameter	Description
`configure voip > sip-definition settings > verify-rcvd-requri` [VerifyRecievedRequestUri]	Enables the device to reject SIP requests (e.g., ACK, BYE, or re-INVITE) whose user part in the Request-URI is different from the user part in the Contact header of the last sent SIP request. ■ [0] = (Default) Disable. Even if the user part is different, the device accepts the SIP request. ■ [1] = Enable. If the user part in the Contact header of the previous SIP request is different to the user part in the Request-URI for in-dialog requests, the device rejects the SIP request. A BYE request is responded with a SIP 481, a re-INVITE request is responded with a SIP 404, and an ACK request is ignored. ■ [2] = If the user part in the Contact header of the previous SIP request is different to the user part in the Request-URI for dialog-initiating INVITE requests, the device rejects the SIP request. ■ Verify dialog-initiating INVITE for all required conditions (Via, Source IP and user in Request-URI) ■ [3] = Verify dialog-initiating INVITE and in-dialog requests. The VerifyRecievedRequestUri parameter functions together with the RegistrarProxySetID parameter, as follows: ■ **Handling Dialog-Initiating INVITEs:** If the VerifyRecievedRequestUri parameter is configured to [2] or [3] and the RegistrarProxySetID parameter is configured to some Proxy Set, the device accepts dialog-initiating INVITE requests received from the registrar at which the Accounts (configured in the Accounts table) are registered. For dialog-initiating INVITE requests received from the registrar on a specific SIP Interface, the following rules apply (listed according to priority): ✓ The top-most Via header must contain a host-resolved IP address of the registrar; otherwise, the device drops the INVITE request. ✓ The source IP address must be the same as the IP address of the registrar; otherwise, the

Parameter	Description
	device rejects the requests and sends a SIP 403 (Forbidden) response to the registrar. ✓ The user part, specified in the Request-URI header, must be identical to the Contact user part configured for the associated Account, and the Account must be registered. Otherwise, the device rejects the request with a SIP 404 (Not Found) response. If the RegistrarProxySetID parameter is not configured or no Accounts are configured, the device accepts the dialog-initiating INVITE request. ■ **Handling In-dialog Requests:** If the VerifyRecievedRequestUri parameter is configured to [1] or [3], for all incoming in-dialog requests (including ACK and CANCEL), the device checks if the Request-URI user part matches the remote Contact user part (i.e., the Contact user configured for the Account). If there is no match, the device rejects the request and sends a SIP 481 response for requests such as BYE and CANCEL, or a SIP 404 for other requests, and for ACK it does not send any response.
[RegistrarProxySetID]	Defines a Proxy Set for the registrar. The parameter functions together with the VerifyRecievedRequestUri parameter. For more information, see the description of the VerifyRecievedRequestUri parameter. The default value is -1 (not defined). **Note:** This setting assumes that the SIP Interface has only one registrar.
'Max Number of Active Calls' `configure voip > sip-definition settings > max-nb-of--act-calls` [MaxActiveCalls]	Defines the maximum number of simultaneous active calls supported by the device. If the maximum number of calls is reached, new calls are not established. The valid range is 1 to the maximum number of supported channels. The default value is the maximum available channels (i.e., no restriction on the maximum number of calls).
'Enable Early Media'	Global parameter enabling the Early Media feature for

Parameter	Description
`early-media` [EnableEarlyMedia]	sending media (e.g., ringing) before the call is established. You can also configure this feature per specific calls, using IP Profiles (IpProfile_EnableEarlyMedia). For a detailed description of the parameter and for configuring the feature, see Configuring IP Profiles. **Note:** If the feature is configured for a specific profile, the settings of the global parameter is ignored for calls associated with the profile.
[RemoveToTagInFailureResponse]	Determines whether the device removes the 'to' header tag from final SIP failure responses to INVITE transactions. ■ [0] = (Default) Do not remove tag. ■ [1] = Remove tag.
'Fax Signaling Method' `fax-sig-method` [IsFaxUsed]	Global parameter defining the SIP signaling method for establishing and transmitting a fax session when the device detects a fax. You can also configure this feature per specific calls, using IP Profiles (IpProfile_IsFaxUsed). For a detailed description of the parameter, see Configuring IP Profiles . **Note:** If this feature is configured for a specific IP Profile, the settings of this global parameter is ignored for calls associated with the IP Profile.
`fax-vbd-behvr` [FaxVBDBehavior]	Determines the device's fax transport behavior when G.711 VBD coder is negotiated at call start. ■ [0] = (Default) If the device is configured with a VBD coder (see the CodersGroup parameter) and is negotiated OK at call start, then both fax and modem signals are sent over RTP using the bypass payload type (and no mid-call VBD or T.38 Re-INVITEs occur). ■ [1] = If the IsFaxUsed parameter is set to 1, the channel opens with the FaxTransportMode parameter set to 1 (relay). This is required to detect mid-call fax tones and to send T.38 Re-INVITE messages upon fax detection. If the remote party supports T.38, the fax is relayed over T.38.

Parameter	Description
	Note: ■ If VBD coder negotiation fails at call start and if the IsFaxUsed parameter is set to 1 (or 3), then the channel opens with the FaxTransportMode parameter set to 1 (relay) to allow future detection of fax tones and sending of T.38 Re-INVITES. In such a scenario, the FaxVBDBehavior parameter has no effect. ■ This feature can be used only if the remote party supports T.38 fax relay; otherwise, the fax fails.
[NoAudioPayloadType]	Defines the payload type of the outgoing SDP offer. The valid value range is 96 to 127 (dynamic payload type). The default is 0 (i.e. NoAudio is not supported). For example, if set to 120, the following is added to the INVITE SDP: `a=rtpmap:120 NoAudio/8000\r\n` Note: For incoming SDP offers, NoAudio is always supported.
'SIP Transport Type' `configure voip > sip-definition settings > app-sip-transport-type` [SIPTransportType]	Determines the default transport layer for outgoing SIP calls initiated by the device. ■ [0] **UDP** (default) ■ [1] **TCP** ■ [2] **TLS (SIPS)** **Note:** ■ It's recommended to use TLS for communication with a SIP Proxy and not for direct device-to-device communication. ■ For received calls (i.e., incoming), the device accepts all these protocols.
'Display Default SIP Port' `configure voip > sip-definition settings > display-default-sip-port` [DisplayDefaultSIPPort]	Enables the device to add the default SIP port 5060 (UDP/TCP) or 5061 (TLS) to outgoing messages that are received without a port. This condition also applies to manipulated messages where the resulting message has no port number. The device adds the default port number to the following SIP headers: Request-Uri, To, From, P-Asserted-Identity, P-Preferred-Identity, and P-

Parameter	Description
	Called-Party-ID. If the message is received with a port number other than the default, for example, 5070, the port number is not changed. An example of a SIP From header with the default port is shown below: `From: <sip:+4000@10.8.4.105:5060 ;user=phone>;tag=f25419a96a;epid=009F AB8F3E` ■ [0] **Disable** (default) ■ [1] **Enable**
'SIPS' `configure voip > sip-definition settings > enable-sips` [EnableSIPS]	Enables secured SIP (SIPS URI) connections over multiple hops. ■ [0] **Disable** (default) ■ [1] **Enable** When the SIPTransportType parameter is set to 2 (i.e., TLS) and the parameter EnableSIPS is disabled, TLS is used for the next network hop only. When the parameter SIPTransportType is set to 2 or 1 (i.e., TCP or TLS) and EnableSIPS is enabled, TLS is used through the entire connection (over multiple hops). **Note:** If the parameter is enabled and the parameter SIPTransportType is set to 0 (i.e., UDP), the connection fails.
'TCP/TLS Connection Reuse' `tcp-conn-reuse` [EnableTCPConnectionReuse]	Enables the reuse of an established TCP or TLS connection between the device and a SIP user agent (UA) for subsequent SIP requests sent to the UA. Any new out-of-dialog requests (e.g., INVITE or REGISTER) use the same secured connection. One of the benefits of enabling the parameter is that it may improve performance by eliminating the need for additional TCP/TLS handshakes with the UA, allowing sessions to be established rapidly. ■ [0] Disable = The device uses a new TCP or TLS connection with the UA. ■ [1] Enable = (Default) The device uses the same TCP or TLS connection for all SIP requests with the UA. **Note:**

Parameter	Description
	■ For SIP responses, the device always uses the same TCP/TLS connection, regardless of the parameter settings.
`configure voip > sip-definition settings > fake-tcp-alias` [FakeTCPalias]	Enables the re-use of the same TCP/TLS connection for sessions with the same user, even if the "alias" parameter is not present in the SIP Via header of the first INVITE. ■ [0] **Disable** = (Default) TCP/TLS connection reuse is done only if the "alias" parameter is present in the Via header of the first INVITE (according to RFC 5923). ■ [1] **Enable** **Note:** To enable TCP/TLS connection re-use, set the [EnableTCPConnectionReuse] parameter to 1.
'Reliable Connection Persistent Mode' `configure voip > sip-definition settings > reliable-conn-persistent` [ReliableConnectionPersistentMode]	Enables setting of all TCP/TLS connections as persistent and therefore, not released. ■ [0] **Disable** = (Default) All TCP connections (except those that are set to a proxy IP) are released if not used by any SIP dialog\transaction and there are no registered users associated with a TCP connection. ■ [1] **Enable** = TCP connections to all destinations are persistent and not released unless the device reaches 70% of its maximum TCP resources. While trying to send a SIP message connection, reuse policy determines whether live connections to the specific destination are re-used. Persistent TCP connection ensures less network traffic due to fewer setting up and tearing down of TCP connections and reduced latency on subsequent requests due to avoidance of initial TCP handshake. For TLS, persistent connection may reduce the number of costly TLS handshakes to establish security associations, in addition to the initial TCP connection set up. **Note:** If the destination is a Proxy server, the TCP/TLS connection is persistent regardless of the settings of the parameter.

Parameter	Description
'TCP Timeout' `configure voip > sip-definition settings > tcp-timeout` [SIPTCPTimeout]	Defines the Timer B (INVITE transaction timeout timer) and Timer F (non-INVITE transaction timeout timer), as defined in RFC 3261, when the SIP transport type is TCP. The valid range is 0 to 60 sec. The default is 0, which means that the parameter's value is set to 64 multiplied by the value of the [SipT1Rtx] parameter. For example, if you configure [SipT1Rtx] to 500 msec (0.5 sec) and leave the [SIPTCPTimeout] parameter at its default value (0), the actual value of [SIPTCPTimeout] is 32 sec (64 x 0.5 sec).
'SIP Destination Port' `configure voip > sip-definition settings > sip-dst-port` [SIPDestinationPort]	Defines the SIP destination port for sending initial SIP requests. The valid range is 1 to 65534. The default port is 5060. **Note:** SIP responses are sent to the port specified in the Via header.
'Use Tel URI for Asserted Identity' `configure voip > sip-definition settings > uri-for-assert-id` [UseTelURIForAssertedID]	Defines the format of the URI in the P-Asserted-Identity and P-Preferred-Identity headers. ■ [0] **Disable** = (Default) The format is 'sip:'. ■ [1] **Enable** = The format is 'tel:'.
`configure voip > sip-definition settings > p-preferred-id-list` [PPreferredIdListMode]	Defines the number of P-Preferred-Identity SIP headers included in the outgoing SIP message when the header contains multiple values. ■ [0] = (Default) The device includes multiple P-Preferred-Identity SIP headers in the outgoing message, for example: ✓ Incoming message containing a P-Preferred-Identity header with multiple values: `P-Preferred-Identity: <sip:someone@test.org>,<tel:+123456789>` ✓ Outgoing message sent with multiple P-Preferred-Identity headers, each with a value:

Parameter	Description
	P-Preferred-Identity: <sip:someone@test.org> P-Preferred-Identity: <tel:+123456789> ■ [1] = The device includes only one P-Preferred-Identity header in the outgoing message, for example: ✓ Incoming message containing multiple P-Preferred-Identity headers: P-Preferred-Identity: <sip:someone@test.org> P-Preferred-Identity: <tel:+123456789> ✓ Outgoing message sent with a single P-Preferred-Identity header containing the multiple values: P-Preferred-Identity: <sip:someone@test.org>,<tel:+123456789> **Note:** ■ If more than two P-Preferred-Identity headers are received in the incoming message, the device keeps the first two headers (if configured to 0) or the first header (if configured to 1), and removes the others in the outgoing message.
'GRUU' `configure voip > sbc settings > enable-gruu` [EnableGRUU]	Determines whether the Globally Routable User Agent URIs (GRUU) mechanism is used, according to RFC 5627. This is used for obtaining a GRUU from a registrar and for communicating a GRUU to a peer within a dialog. ■ [0] **Disable** (default) ■ [1] **Enable** A GRUU is a SIP URI that routes to an instance-specific

Parameter	Description

	UA and can be reachable from anywhere. There are a number of contexts in which it is desirable to have an identifier that addresses a single UA (using GRUU) rather than the group of UA's indicated by an Address of Record (AOR). For example, in call transfer where user A is talking to user B, and user A wants to transfer the call to user C. User A sends a REFER to user C:
	`REFER sip:C@domain.com SIP/2.0`
	`From: sip:A@domain.com;tag=99asd`
	`To: sip:C@domain.com`
	`Refer-To: (URI that identifies B's UA)`
	The Refer-To header needs to contain a URI that user C can use to place a call to user B. This call needs to route to the specific UA instance that user B is using to talk to user A. User B should provide user A with a URI that has to be usable by anyone. It needs to be a GRUU.
	■ Obtaining a GRUU: The mechanism for obtaining a GRUU is through registrations. A UA can obtain a GRUU by generating a REGISTER request containing a Supported header field with the value "gruu". The UA includes a "+sip.instance" Contact header parameter of each contact for which the GRUU is desired. This Contact parameter contains a globally unique ID that identifies the UA instance. The global unique ID is created from one of the following:
	✓ If the REGISTER is per the device's client (endpoint), it is the MAC address concatenated with the phone number of the client.
	✓ If the REGISTER is per device, it is the MAC address only.
	✓ When using TP, "User Info" can be used for registering per endpoint. Thus, each endpoint can get a unique id – its phone number. The globally unique ID in TP is the MAC address concatenated with the phone number of the endpoint.

Parameter	Description
	If the remote server doesn't support GRUU, it ignores the parameters of the GRUU. Otherwise, if the remote side also supports GRUU, the REGISTER responses contain the "gruu" parameter in each Contact header. The parameter contains a SIP or SIPS URI that represents a GRUU corresponding to the UA instance that registered the contact. The server provides the same GRUU for the same AOR and instance-id when sending REGISTER again after registration expiration. RFC 5627 specifies that the remote target is a GRUU target if its' Contact URL has the "gr" parameter with or without a value. ■ Using GRUU: The UA can place the GRUU in any header field that can contain a URI. It must use the GRUU in the following messages: INVITE request, its 2xx response, SUBSCRIBE request, its 2xx response, NOTIFY request, REFER request and its 2xx response.

'User-Agent Information'
`configure voip > sip-definition settings > user-agent-info`
[UserAgentDisplayInfo]
	Defines the string that is used in the SIP User-Agent and Server response headers. When configured, the string <UserAgentDisplayInfo value>/software version' is used, for example:
	`User-Agent: myproduct/7.20A.258.980`
	If not configured, the default string, "<product-name>/<<software version>>" is used, for example:
	`User-Agent: AudioCodes-Sip-Gateway/<swver>`
	The maximum string length is 50 characters.
	Note: The software version number and preceding forward slash (/) cannot be modified. Therefore, it is recommended not to include a forward slash in the parameter's value (to avoid two forward slashes in the SIP header, which may cause problems).

'SDP Session Owner' `configure voip > sip-definition settings > sdp-session-owner` [SIPSDPSessionOwner]	Defines the value of the Owner line ('o' field) in outgoing SDP messages. The valid range is a string of up to 39 characters. The default is "AudioCodesGW". For example: `o=AudioCodesGW 1145023829 1145023705 IN IP4 10.33.4.126`

Parameter	Description
	Note: The parameter is applicable only when the device creates a new SIP message (and SDP) such as when the device plays a ringback tone. The parameter is not applicable to SIP messages that the device receives from one end and sends to another (i.e., does not modify the SDP's 'o' field).
`configure voip > sip-definition settings > sdp-ver-nego` [EnableSDPVersionNegotiation]	Enables the device to ignore new SDP re-offers (from the media negotiation perspective) in certain scenarios (such as session expires). According to RFC 3264, once an SDP session is established, a new SDP offer is considered a new offer only when the SDP origin value is incremented. In scenarios such as session expires, SDP negotiation is irrelevant and thus, the origin field is not changed. Even though some SIP devices don't follow this behavior and don't increment the origin value even in scenarios where they want to re-negotiate, the device can assume that the remote party operates according to RFC 3264, and in cases where the origin field is not incremented, the device does not re-negotiate SDP capabilities. ■ [0] Disable = (Default) The device negotiates any new SDP re-offer, regardless of the origin field. ■ [1] Enable = The device negotiates only an SDP re-offer with an incremented origin field.
'Subject' `configure voip > sip-definition settings > usr-def-subject` [SIPSubject]	Defines the Subject header value in outgoing INVITE messages. If not specified, the Subject header isn't included (default). The maximum length is up to 50 characters.
'Multiple Packetization Time Format' `configure voip > sip-definition settings > mult-ptime-format` [MultiPtimeFormat]	Determines whether the 'mptime' attribute is included in the outgoing SDP. ■ [0] **None** = (Default) Disabled. ■ [1] **PacketCable** = Includes the 'mptime' attribute in the outgoing SDP - PacketCable-defined format. The mptime' attribute enables the device to define a separate packetization period for each negotiated

Parameter	Description
	coder in the SDP. The 'mptime' attribute is only included if the parameter is enabled even if the remote side includes it in the SDP offer. Upon receipt, each coder receives its 'ptime' value in the following precedence: from 'mptime' attribute, from 'ptime' attribute, and then from default value.
`configure voip > sip-definition settings > enable-ptime` [EnablePtime]	Determines whether the 'ptime' attribute is included in the SDP. ■ [0] = Remove the 'ptime' attribute from SDP. ■ [1] = (Default) Include the 'ptime' attribute in SDP.
'3xx Behavior' `3xx-behavior` [3xxBehavior]	Determines the device's behavior regarding call identifiers when a 3xx response is received for an outgoing INVITE request. The device can use the same call identifiers (Call-ID, To, and From tags) or change them in the new initiated INVITE. ■ [0] **Forward** = (Default) Use different call identifiers for a redirected INVITE message. ■ [1] **Redirect** = Use the same call identifiers in the new INVITE as the original call.
[RetryAfterMode]	Defines the device's behavior when it receives a SIP 503 (Service Unavailable) containing a Retry-After header, in response to a SIP message (e.g., REGISTER) sent to the proxy server. In certain scenarios (depending on the value of this parameter), the device considers the proxy as offline (down) for the number of seconds specified in the Retry-After header. During this timeout, the device does not send any SIP messages to the proxy. This condition is indicated in the Syslog message as "server is now Unavailable - setting Retry-After timer to x secs". ■ [1] = (Default) Handle Locally. The device considers the proxy as offline regardless of the type of SIP message sent to the proxy for which the 503 response was received. ■ [0] = Transparent. The behavior depends on the type of SIP message sent to the proxy for which the 503 response was received:

Parameter	Description
	✓ SIP OPTIONS message: The device considers the proxy as offline. ✓ SIP REGISTER message generated (created) by the device: The device does not send REGISTER messages to the proxy for this **specific** registration process (i.e., Accounts table or User Information table) during the timeout specified in the Retry-After header of the 503 response. However, the device considers the proxy as online and therefore, it continues sending traffic of other entities to the proxy. ✓ All other SIP dialogs (e.g., INVITE): The device ignores the Retry-After header and forwards the 503 response transparently to the other user agent.
'Retry-After Time' `configure voip > sip-definition settings > retry-aftr-time` [RetryAfterTime]	Defines the time (in seconds) used in the Retry-After header when a 503 (Service Unavailable) response is generated by the device. The time range is 0 to 3,600. The default is 0.
'Fake Retry After' `fake-retry-after` [FakeRetryAfter]	Determines whether the device, upon receipt of a SIP 503 response without a Retry-After header, behaves as if the 503 response included a Retry-After header and with the period (in seconds) specified by the parameter. ■ [0] Disable (default) ■ Any positive value (in seconds) for defining the period When enabled, this feature allows the device to operate with Proxy servers that do not include the Retry-After SIP header in SIP 503 (Service Unavailable) responses to indicate an unavailable service. The Retry-After header is used with the 503 (Service Unavailable) response to indicate how long the service is expected to be unavailable to the requesting SIP client. The device maintains a list of available proxies, by using the Keep-Alive mechanism. The device checks the availability of proxies by sending SIP OPTIONS every keep-alive timeout to all proxies.

Parameter	Description
	If the device receives a SIP 503 response to an INVITE, it also marks that the proxy is out of service for the defined "Retry-After" period.
'P-Associated-URI Header' `p-associated-uri-hdr` [EnablePAssociatedURIHeader]	Determines the device usage of the P-Associated-URI header. This header can be received in 200 OK responses to REGISTER requests. When enabled, the first URI in the P-Associated-URI header is used in subsequent requests as the From/P-Asserted-Identity headers value. ■ [0] **Disable** (default) ■ [1] **Enable** **Note:** P-Associated-URIs in registration responses is handled only if the device is registered per endpoint (using the User Information file).
'Source Number Preference' `configure voip > sip-definition settings > src-nb-preference` [SourceNumberPreference]	Defines the SIP header from which the source (calling) number is obtained in incoming INVITE messages. ■ If not configured or if any string other than "From" or "Pai2" is configured, the calling number is obtained from a specific header using the following logic: a. P-Preferred-Identity header. b. If the above header is not present, then the first P-Asserted-Identity header is used. c. If the above header is not present, then the Remote-Party-ID header is used. d. If the above header is not present, then the From header is used. ■ "**From**" = The calling number is obtained from the From header. ■ "**Pai2**" = The calling number is obtained using the following logic: e. If a P-Preferred-Identity header is present, the number is obtained from it. f. If no P-Preferred-Identity header is present and two P-Asserted-Identity headers are present, the number is obtained from the second P-

Parameter	Description
	Asserted-Identity header. g. If only one P-Asserted-Identity header is present, the calling number is obtained from it. **Note:** ■ The "From" and "Pai2" values are not case-sensitive. ■ Once a URL is selected, all the calling party parameters are set from this header. If P-Asserted-Identity is selected and the Privacy header has the value 'id', the calling number is assumed restricted.
'Reason Header' `configure voip > sip-definition settings > reason-header` [EnableReasonHeader]	Enables the usage of the SIP Reason header. ■ [0] **Disable** ■ [1] **Enable** (default)
'Gateway Name' `configure voip > sip-definition settings > gw-name` [SIPGatewayName]	Defines a name for the device (e.g., device123.com), which is used as the host part for the SIP URI in the From header for outgoing messages. If not configured, the device's IP address is used instead (default). The valid value is a string of up to 100 characters. By default, no value is defined. **Note:** ■ Ensure that the parameter value is the one with which the proxy has been configured with to identify the device. ■ If you enable keep-alive by SIP OPTIONS messages with the proxy (see the [ProxySet_EnableProxyKeepAlive] parameter), you can configure, using the [UseGatewayNameForOptions] parameter, if the device's IP address, the proxy's IP address, or the device's name (configured by the [SIPGatewayName] parameter) is used in the keep-alive SIP OPTIONS messages (host part of the Request-URI). ■ The parameter can also be configured for an IP Group (in the IP Groups table).
`configure voip > sip-`	Determines the device's response to an incoming SDP

Parameter	Description
`definition settings > zero-sdp-behavior` [ZeroSDPHandling]	that includes an IP address of 0.0.0.0 in the SDP's Connection Information field (i.e., "c=IN IP4 0.0.0.0"). ■ [0] = (Default) Sets the IP address of the outgoing SDP's c= field to 0.0.0.0. ■ [1] = Sets the IP address of the outgoing SDP c= field to the IP address of the device. If the incoming SDP doesn't contain the "a=inactive" line, the returned SDP contains the "a=recvonly" line.
'Delayed Offer' `configure voip > sip-definition settings > delayed-offer` [EnableDelayedOffer]	Determines whether the device sends the initial INVITE message with or without an SDP. Sending the first INVITE without SDP is typically done by clients for obtaining the far-end's full list of capabilities before sending their own offer. (An alternative method for obtaining the list of supported capabilities is by using SIP OPTIONS, which is not supported by every SIP agent.) ■ [0] **Disable** = (Default) The device sends the initial INVITE message with an SDP. ■ [1] **Enable** = The device sends the initial INVITE message without an SDP.
[SIPDigestAuthorizationURIMode]	Defines whether the device includes or excludes URI parameters for the Digest URI in the SIP Proxy-Authorization or Authorization headers of the request that the device sends in reply to a received SIP 401 (Unauthorized) or 407 (Proxy Authentication Required) response. Below shows an example of a request with an Authorization header containing a Digest URI (shown in bold): `Authorization: Digest username="alice at AudioCodes .com",realm="AudioCodes .com",nonce="",response="",` **`uri="sip:AudioCodes.com"`** ■ [0] = (Default) The device sends the request without a Digest URI. ■ [1] = The device sends the request with a Digest

Parameter	Description
	URI, which is set to the same value as the URI in the original Request-URI.
`configure voip > sip-definition settings > crypto-life-time-in-sdp` [DisableCryptoLifeTimeInSDP]	Enables the device to send "a=crypto" lines without the lifetime parameter in the SDP. For example, if the SDP contains "a=crypto:12 AES_CM_128_HMAC_SHA1_80 inline:hhQe10yZRcRcpIFPkH5xYY9R1de37ogh9G1MpvNp|2^31", it removes the lifetime parameter "2^31". ■ [0] Disable (default) ■ [1] Enable
'Contact Restriction' `contact-restriction` [EnableContactRestriction]	Determines whether the device sets the Contact header of outgoing INVITE requests to 'anonymous' for restricted calls. ■ [0] **Disable** (default) ■ [1] **Enable**
`configure voip > sip-definition settings > use-aor-in-refer-to-header` [UseAORInReferToHeader]	Defines the source for the SIP URI set in the Refer-To header of outgoing REFER messages. ■ [0] = (Default) Use SIP URI from Contact header of the initial call. ■ [1] = Use SIP URI from To/From header of the initial call.
'User-Information Usage' `configure voip > sip-definition settings > user-inf-usage` [EnableUserInfoUsage]	Enables the usage of the User Information, which is loaded to the device in the User Information Auxiliary file. For more information on User Information, see User Information File. ■ [0] **Disable** (default) ■ [1] **Enable** **Note:** For the parameter to take effect, a device reset is required.
`configure voip > sip-definition settings > handle-reason-header` [HandleReasonHeader]	Determines whether the device uses the value of the incoming SIP Reason header for Release Reason mapping. ■ [0] = Disregard Reason header in incoming SIP messages.

Parameter	Description
	■ [1] = (Default) Use the Reason header value for Release Reason mapping.
[EnableSilenceSuppInSDP]	Determines the device's behavior upon receipt of SIP Re-INVITE messages that include the SDP's 'silencesupp:off' attribute. ■ [0] = (Default) Disregard the 'silecesupp' attribute. ■ [1] = Handle incoming Re-INVITE messages that include the 'silencesupp:off' attribute in the SDP as a request to switch to the Voice-Band-Data (VBD) mode. In addition, the device includes the attribute 'a=silencesupp:off' in its SDP offer. **Note:** The parameter is applicable only if the G.711 coder is used.
`configure voip > sip-definition settings > rport-support` [EnableRport]	Enables the usage of the 'rport' parameter in the Via header. ■ [0] = Disabled (default) ■ [1] = Enabled The device adds an 'rport' parameter to the Via header of each outgoing SIP message. The first Proxy that receives this message sets the 'rport' value of the response to the actual port from where the request was received. This method is used, for example, to enable the device to identify its port mapping outside a NAT. If the Via header doesn't include the 'rport' parameter, the destination port of the response is obtained from the host part of the Via header. If the Via header includes the 'rport' parameter without a port value, the destination port of the response is the source port of the incoming request. If the Via header includes 'rport' with a port value (e.g., rport=1001), the destination port of the response is the port indicated in the 'rport' parmeter.
[EnableRekeyAfter181]	Enables the device to send a re-INVITE with a new (different) SRTP key (in the SDP) if a SIP 181 response is received ("call is being forwarded"). The re-INVITE is sent immediately upon receipt of the 200 OK (when the call is answered).

Parameter	Description
	■ [0] = Disable (default) ■ [1] = Enable **Note:** The parameter is applicable only if SRTP is used.
`configure voip > sip-definition settings > number-of-active-dialogs` [NumberOfActiveDialogs]	Defines the maximum number of concurrent, outgoing SIP REGISTER dialogs. The parameter is used to control the registration rate. The valid range is 1 to 20. The default is 20. **Note:** ■ Once a 200 OK is received in response to a REGISTER message, the REGISTER message is not considered in this maximum count limit. ■ The parameter applies only to outgoing REGISTER messages (i.e., incoming is unlimited).
'Network Node ID' `configure voip > sip-definition settings > net-node-id` [NetworkNodeId]	Defines the Network Node Identifier of the device for Avaya UCID. The valid value range is1 to 0x7FFF. The default is 0. **Note:** ■ To use this feature, you must set the parameter to any value other than 0. ■ To enable the generation by the device of the Avaya UCID value and adding it to the outgoing INVITE sent to the IP Group (Avaya entity), use the IP Groups table's parameter 'UUI Format'.
'Enable Microsoft Extension' `configure voip > sip-definition settings > microsoft-ext` [EnableMicrosoftExt]	Enables the modification of the called and calling number for numbers received with Microsoft's proprietary "ext=xxx" parameter in the SIP INVITE URI user part. Microsoft Office Communications Server sometimes uses this proprietary parameter to indicate the extension number of the called or calling party. ■ [0] **Disable** (default) ■ [1] **Enable** For example, if a calling party makes a call to telephone number 622125519100 Ext. 104, the device receives the SIP INVITE (from Microsoft's application) with the URI user part as INVITE

Parameter	Description
	sip:622125519100;ext=104@10.1.1.10 (or INVITE tel:622125519100;ext=104). If the parameter EnableMicrosofExt is enabled, the device modifies the called number by adding an "e" as the prefix, removing the "ext=" parameter, and adding the extension number as the suffix (e.g., **e**622125519100**104**). Once modified, the device can then manipulate the number further, using the Number Manipulation tables to leave only the last 3 digits (for example) for sending to a PBX.
`configure voip > sip-definition settings > sip-uri-for-diversion-header` [UseSIPURIForDiversionHeader]	Defines the URI format in the SIP Diversion header. ■ [0] = 'tel:' (default) ■ [1] = 'sip:'
`configure voip > sip-definition settings > 100-to-18x-timeout` [TimeoutBetween100And18x]	Defines the timeout (in msec) between receiving a 100 Trying response and a subsequent 18x response. If a 18x response is not received within this timeout period, the call is disconnected. The valid range is 0 to 180,000 (i.e., 3 minutes). The default is 32000 (i.e., 32 sec).
`configure voip > sip-definition settings > ignore-remote-sdp-mki` [IgnoreRemoteSDPMKI]	Determines whether the device ignores the Master Key Identifier (MKI) if present in the SDP received from the remote side. ■ [0] = (Default) Disable ■ [1] = Enable
`configure voip > sip-definition settings > sdp-ecan-frmt` [SDPEcanFormat]	Defines the echo canceller format in the outgoing SDP. The 'ecan' attribute is used in the SDP to indicate the use of echo cancellation. ■ [0] = (Default) The 'ecan' attribute appears on the 'a=gpmd' line. ■ [1] = The 'ecan' attribute appears as a separate attribute. ■ [2] = The 'ecan' attribute is not included in the SDP. ■ [3] = The 'ecan' attribute and the 'vbd' parameter are not included in the SDP.

Parameter	Description
	Note: The parameter is applicable only when the IsFaxUsed parameter is set to 2, and for re-INVITE messages generated by the device as result of modem or fax tone detection.
'First Call Ringback Tone ID' `configure voip > sip-definition settings > 1st-call-rbt-id` [FirstCallRBTId]	Defines the index of the first ringback tone in the CPT file. This option enables an Application server to request the device to play a distinctive ringback tone to the calling party according to the destination of the call. The tone is played according to the Alert-Info header received in the 180 Ringing SIP response (the value of the Alert-Info header is added to the value of the parameter). The valid range is -1 to 1,000. The default is -1 (i.e., play standard ringback tone). **Note:** ■ It is assumed that all ringback tones are defined in sequence in the CPT file. ■ In case of an MLPP call, the device uses the value of the parameter plus 1 as the index of the ringback tone in the CPT file (e.g., if this value is set to 1, then the index is 2, i.e., 1 + 1).
'Presence Publish IP Group ID' [PresencePublishIPGroupId]	Assigns the IP Group (by ID) configured for the Skype for Business Server (presence server). This is where the device sends SIP PUBLISH messages to notify of changes in presence status of Skype for Business users when making and receiving calls using third-party endpoint devices. For more information on integration with Microsoft presence, see Microsoft Skype for Business Presence of Third-Party Endpoints.
'Microsoft Presence Status' [EnableMSPresence]	Enables the device to notify (using SIP PUBLISH messages) Skype for Business Server (presence server) of changes in presence status of Skype for Business users when making and receiving calls using third-party endpoint devices. ■ [0] **Disable** (default) ■ [1] **Enable** For more information on integration with Microsoft

Parameter	Description
	presence, see Microsoft Skype for Business Presence of Third-Party Endpoints.
'Media IP Version Preference' media-ip-ver-pref [MediaIPVersionPreference]	Global parameter that defines the preferred RTP media IP addressing version (IPv4 or IPv6) for outgoing SIP calls. You can also configure this feature per specific calls, using IP Profiles (IpProfile_ MediaIPVersionPreference). For a detailed description of the parameter and for configuring this feature in the IP Profiles table, see Configuring IP Profiles.
configure voip > message settings > inbound-map-set [GWInboundManipulationSet]	Assigns a Manipulation Set ID for manipulating incoming responses of requests that the device initiates. The Manipulation Set is defined using the MessageManipulations parameter. By default, no manipulation is done (i.e. Manipulation Set ID is set to -1). For more information, see Configuring SIP Message Manipulation on page 634.
configure voip > message settings > outbound-map-set [GWOutboundManipulationSet]	Assigns a Manipulation Set ID for manipulating outgoing requests that the device initiates. The Manipulation Set is defined using the MessageManipulations parameter. By default, no manipulation is done (i.e. Manipulation Set ID is set to -1). For more information, see Configuring SIP Message Manipulation on page 634.
'WebSocket Keep-Alive Period' configure voip > sip-definition settings > websocket-keepalive [WebSocketProtocolKeepAlivePeriod]	Defines how often (in seconds) the device sends ping messages (keep alive) to check whether the WebSocket session with the Web client is still connected. The valid value is 5 to 2000000. The default is 0 (i.e., ping messages are not sent). For more information on WebSocket, see SIP over WebSocket. **Note:** ■ The device always replies to WebSocket ping control messages with pong messages.

Parameter	Description
Out-of-Service (Busy Out) Parameters **Retransmission Parameters**	
'SIP T1 Retransmission Timer' `configure voip > sip-definition settings > t1-re-tx-time` [SipT1Rtx]	Defines the time interval (in msec) between the first transmission of a SIP message and the first retransmission of the same message. The default is 500. **Note:** The time interval between subsequent retransmissions of the same SIP message starts with SipT1Rtx. For INVITE requests, it is multiplied by two for each new retransmitted message. For all other SIP messages, it is multiplied by two until SipT2Rtx. For example, assuming SipT1Rtx = 500 and SipT2Rtx = 4000: ■ The first retransmission is sent after 500 msec. ■ The second retransmission is sent after 1000 (2*500) msec. ■ The third retransmission is sent after 2000 (2*1000) msec. ■ The fourth retransmission and subsequent retransmissions until [SIPMaxRtx] are sent after 4000 (2*2000) msec.
'SIP T2 Retransmission Timer' `configure voip > sip-definition settings > t2-re-tx-time` [SipT2Rtx]	Defines the maximum interval (in msec) between retransmissions of SIP messages (except for INVITE requests). The default is 4000. **Note:** The time interval between subsequent retransmissions of the same SIP message starts with SipT1Rtx and is multiplied by two until SipT2Rtx.
'SIP Maximum RTX' `configure voip > sip-definition settings > sip-max-rtx` [SIPMaxRtx]	Defines the maximum number of UDP transmissions of SIP messages (first transmission plus retransmissions). The range is 1 to 30. The default is 7.
'Number of RTX Before Hot-Swap' `configure voip > sip-`	Defines the number of retransmitted INVITE/REGISTER messages before the call is routed (hot swap) to another Proxy/Registrar.

Parameter	Description
`definition proxy-and-registration > nb-of-rtx-b4-hot-swap` [HotSwapRtx]	The valid range is 1 to 30. The default is 3. For example, if configured to 3 and no response is received from an IP destination, the device attempts another three times to send the call to the IP destination. If still unsuccessful, it attempts to redirect the call to another IP destination. **Note:** The parameter is also used for alternative routing (see Alternative Routing Based on IP Connectivity.
`configure voip > sip-definition settings > message-policy-reject-response-type` [MessagePolicyRejectResponseType]	Defines the SIP response code that the device sends when it rejects an incoming SIP message due to a matched Message Policy in the Message Policies table, whose 'Send Reject' (MessagePolicy_SendRejection) parameter is configured to Policy Reject [0]. The default is 400 "Bad Request". To configure Message Policies, see Configuring SIP Message Policy Rules.
[ENUMAllowNonDigits]	Defines if non-digits can be included in ENUM queries sent by the device to an ENUM server for retrieving a SIP URI address for an E.164 telephone number (destination). ■ [0] = (Default) Disable – non-digits are not accepted in ENUM queries. For example: 9.2.0.0.3.0.9.3.0.3.0.2.5.3.4.4.2.5.7.7.7.8.My_Domain ■ [1] = Enable – non-digits are accepted in ENUM queries (request). For example: 0.0.0.0.0.2.3.3.3.3.2.2.*.9.9.j.a.k.s.*.j.k.a.n.d.b.j.s.+.My_Domain ENUM queries can be used for IP-to-IP routing with Call Setup Rules (see Configuring SBC IP-to-IP Routing on page 716 and Configuring Call Setup Rules on page 595).
`configure voip > sip-definition settings > preserve-multipart-content-type` [PreserveMultipartContentType]	Defines the device's handling of the SIP Content-Type header's value when the device sends a SIP message that has multiple bodies. ■ [0] = (Default) Disabled. The device sets the type parameter in the Content-Type header to

Parameter	Description
	"multipart/mixed" and adds a unique value to the 'boundary' parameter of the Content-Type header. ■ [1] = Enabled. The device doesn't change the type or boundary parameter of the Content-Type header. For example, if the incoming message contains 'Content-Type: multipart/relative;boundary=<someUniqueValue>', then this is how the Content-Type will be in the outgoing message.

Channel Parameters

This section describes the device's channel parameters.

Voice Parameters

The voice parameters are described in the table below.

Table 60-32:Voice Parameters

Parameter	Description
'Input Gain' `configure voip > media voice > input-gain` [InputGain]	Global parameter defining the pulse-code modulation (PCM) input (received) gain control level (in decibels). You can also configure the feature per specific calls, using IP Profiles (IpProfile_ InputGain) . For a detailed description of the parameter and for configuring the feature, see Configuring IP Profiles. **Note:** If the feature is configured for a specific profile, the settings of the global parameter is ignored for calls associated with the profile.
'Voice Volume' `configure voip > media voice > voice-volume` [VoiceVolume]	Global parameter defining the voice gain control (in decibels). This defines the level of the transmitted signal. You can also configure the feature per specific calls, using IP Profiles (IpProfile_ VoiceVolume) . For a detailed description of the parameter and for configuring the feature, see Configuring IP Profiles.

Parameter	Description
	Note: If the feature is configured for a specific profile, the settings of the global parameter is ignored for calls associated with the profile.
`configure voip > media voice codecs > G726-voice-payload-format` [VoicePayloadFormat]	Determines the bit ordering of the G.726 voice payload format. ■ [0] = (Default) Little Endian ■ [1] = Big Endian **Note:** To ensure high voice quality when using G.726, both communicating ends should use the same endianness format. Therefore, when the device communicates with a third-party entity that uses the G.726 voice coder and voice quality is poor, change the settings of the parameter (between Big Endian and Little Endian).
'Echo Canceler' `configure voip > media voice > echo-canceller-enable` [EnableEchoCanceller]	Global parameter enabling echo cancellation (i.e., echo from voice calls is removed). You can also configure this feature per specific calls, using IP Profiles (IpProfile_ EnableEchoCanceller)) . For a detailed description of the parameter and for configuring the feature, see Configuring IP Profiles . **Note:** If the feature is configured for a specific profile, the settings of this global parameter is ignored for calls associated with the profile.
'Network Echo Suppressor Enable' `configure voip/media voice/acoustic-echo-suppressor-enable` [AcousticEchoSuppressorSupport]	Enables the network Acoustic Echo Suppressor feature on SBC calls. This feature removes echoes and sends only the near-end's desired speech signal to the network (i.e., to the far-end party). ■ [0] **Disable** (default) ■ [1] **Enable** **Note:** For the parameter to take effect, a device reset is required.

Parameter	Description
'Echo Canceller Type' `configure voip/media voice/echo-canceller-type` [EchoCancellerType]	Defines the echo canceller type. ■ [0] **Line echo canceller** = (Default) Echo canceller for Tel side. ■ [1] **Acoustic Echo suppressor - network** = Echo canceller for IP side.
'Attenuation Intensity' `configure voip/media voice/acoustic-echo-suppressor-attenuation-intensity` [AcousticEchoSuppAttenuationIntensity]	Defines the acoustic echo suppressor signals identified as echo attenuation intensity. The valid range is 0 to 3. The default is 0.
'Max ERL Threshold - DB' `configure voip/media voice/acoustic-echo-suppressor-max-ERL` [AcousticEchoSuppMaxERLThreshold]	Defines the acoustic echo suppressor maximum ratio between signal level and returned echo from the phone (in decibels). The valid range is 0 to 60. The default is 10.
'Min Reference Delay x10 msec' `configure voip/media voice/acoustic-echo-suppressor-min-reference-delay` [AcousticEchoSuppMinRefDelayx10ms]	Defines the acoustic echo suppressor minimum reference delay (in 10-ms units). The valid range is 0 to 40. The default is 0.
'Max Reference Delay x10 msec' `configure voip/media voice/acoustic-echo-suppressor-max-reference-delay` [AcousticEchoSuppMaxRefDelayx10ms]	Defines the acoustic echo suppressor maximum reference delay (in 10-ms units). The valid range is 0 to 40. The default is 40 (i.e., 40 x 10 = 400 ms).
`configure voip > media voice > echo-canceller-hybrid-loss` [ECHybridLoss]	Defines the four-wire to two-wire worst-case Hybrid loss, the ratio between the signal level sent to the hybrid and the echo level returning from the hybrid. ■ [0] = (Default) 6 dB ■ [1] = N/A ■ [2] = 0 dB

Parameter	Description
	■ [3] = 3 dB
`configure voip > media voice > echo-canceller-NLP-mode` [ECNLPMode]	Global parameter enabling Non-Linear Processing (NLP) mode for echo cancellation. ■ [0] = (Default) NLP adapts according to echo changes ■ [1] = Disables NLP **Note:** If the feature is configured for a specific Tel Profile, the settings of the global parameter is ignored for calls associated with the Tel Profile.
`configure voip > media voice > echo-canceller-aggressive-NLP` [EchoCancellerAggressiveNLP]	Enables the Aggressive NLP at the first 0.5 second of the call. ■ [0] = Disable ■ [1] = (Default) Enable. The echo is removed only in the first half of a second of the incoming IP signal. **Note:** For the parameter to take effect, a device reset is required.
`configure voip > media RTP-RTCP > number-of-SID-coefficients` [RTPSIDCoeffNum]	Defines the number of spectral coefficients added to an SID packet being sent according to RFC 3389. The valid values are [0] (default), [4], [6], [8] and [10].

Coder Parameters

The coder parameters are described in the table below.

Table 60-33:Coder Parameters

Parameter	Description
'SILK Tx Inband FEC' `configure voip > media settings > silk-tx-inband-fec` [SilkTxInbandFEC]	Enables forward error correction (FEC) for the SILK coder. ■ [0] **Disable** (default) ■ [1] **Enable**

Parameter	Description
'SILK Max Average Bit Rate' `configure voip > media settings > silk-max-average-bitrate` [SilkMaxAverageBitRate]	Defines the maximum average bit rate for the SILK coder. The valid value range is 6,000 to 50,000. The default is 50,000. The SILK coder is Skype's default audio codec used for Skype-to-Skype calls.
'Opus Max Average Bitrate' `configure voip > sip-definition settings > opus-max-avg-bitrate` [OpusMaxAverageBitRate	Defines the maximum average bit rate (in bps) for the Opus coder. The valid value range is 6000 to 50,000. The default is 50,000.
`configure voip > media settings > vbr-coder-header-format` [VBRCoderHeaderFormat]	Determines the format of the RTP header for VBR coders. ■ [0] = (Default) Payload only (no header, TOC, or m-factor) - similar to RFC 3558 Header Free format. ■ [1] = Supports RFC 2658 - 1 byte for interleaving header (always 0), TOC, no m-factor. ■ [2] = Payload including TOC only, allow m-factor. ■ [3] = RFC 3558 Interleave/Bundled format.
`configure voip > media settings > vbr-coder-hangover` [VBRCoderHangover]	Defines the required number of silence frames at the beginning of each silence period when using the VBR coder silence suppression. The range is 0 to 255. The default is 1.
'AMR Payload Format' [AmrOctetAlignedEnable]	Defines the AMR payload format type. ■ [0] **Bandwidth Efficient** ■ [1] **Octet Aligned** (default) **Note:** The AMR payload type can also be configured per Coder Group (see Configuring Coder Groups). The Coder Group configuration overrides the parameter.
`configure voip > media settings > amr-header-format` [AMRCoderHeaderFormat]	Determines the payload format of the AMR header. ■ [0] = Non-standard multiple frames packing in a single RTP frame. Each frame has a CMR and TOC header. ■ [1] = AMR frame according to RFC 3267 bundling.

Parameter	Description
	■ [2] = AMR frame according to RFC 3267 interleaving. ■ [3] = AMR is passed using the AMR IF2 format. **Note:** Bandwidth Efficient mode is not supported; the mode is always Octet-aligned.
'Fax/Modem Bypass Packing Factor' `configure voip > media fax-modem > packing-factor` [FaxModemBypassM]	Defines the number (20 msec) of coder payloads used to generate a fax/modem bypass packet. The valid range is 1, 2, or 3 coder payloads. The default is 1 coder payload.

DTMF Parameters

The dual-tone multi-frequency (DTMF) parameters are described in the table below.

Table 60-34:DTMF Parameters

Parameter	Description
'DTMF Transport Type' `configure voip > media voice > DTMF-transport-type` [DTMFTransportType]	Defines the DTMF transport type. ■ [0] **Mute DTMF** = DTMF digits are removed from the voice stream and are not relayed to remote side. ■ [2] **Transparent DTMF** = DTMF digits remain in the voice stream. ■ [3] **RFC 2833 Relay DTMF** = (Default) DTMF digits are removed from the voice stream and are relayed to the remote side according to RFC 2833. ■ [7] **RFC 2833 Relay Decoder Mute** = DTMF digits are sent according to RFC 2833 and muted when received. **Note:** The parameter is automatically updated if the parameters [FirstTxDTMFOption] or [RxDTMFOption] are configured.
'DTMF Volume' (-31 to 0 dB) `configure voip > media voice > DTMF-volume`	Global parameter defining the DTMF gain control value (in decibels). **Note:** If the feature is configured for a specific Tel Profile, the settings of the global parameter is ignored for calls associated with the Tel Profile.

Parameter	Description
[DTMFVolume]	
DTMF Generation Twist `configure voip > media voice > DTMF-generation-twist` [DTMFGenerationTwist]	Defines the range (in decibels) between the high and low frequency components in the DTMF signal. Positive decibel values cause the higher frequency component to be stronger than the lower one. Negative values cause the opposite effect. For any parameter value, both components change so that their average is constant. The valid range is -10 to 10 dB. The default is 0 dB. **Note:** For the parameter to take effect, a device reset is required.
`inter-digit-interval` [DTMFInterDigitInterval]	Defines the time (in msec) between generated DTMF digits if FirstTxDTMFOption = 1, 2 or 3. The valid range is 0 to 32767. The default is 100.
[DTMFDigitLength]	Defines the time (in msec) for generating DTMF tones if FirstTxDTMFOption = 1, 2 or 3. It also configures the duration that is sent in INFO (Cisco) messages. The valid range is 0 to 32767. The default is 100.
`configure voip > media voice > digit-hangover-time-rx` [RxDTMFHangOverTime]	Defines the Voice Silence time (in msec) after playing DTMF or MF digits that arrive as Relay . Valid range is 0 to 2,000 msec. The default is 1,000 msec.
`configure voip > media voice > digit-hangover-time-tx` [TxDTMFHangOverTime]	Defines the Voice Silence time (in msec) after detecting the end of DTMF or MF digits when the DTMF Transport Type is either Relay or Mute. Valid range is 0 to 2,000 msec. The default is 1,000 msec.
'NTE Max Duration' `configure voip > media voice > telephony-events-max-duration` [NTEMaxDuration]	Defines the maximum time for sending Named Telephony Events / NTEs -- RFC 4733/2833 DTMF relay -- , regardless of the DTMF signal duration on the other side . The range is -1 to 200,000,000 msec. The default is -1 (i.e., NTE stops only upon detection of an End event).

RTP, RTCP and T.38 Parameters

The RTP, RTCP and T.38 parameters are described in the table below.

Table 60-35:RTP/RTCP and T.38 Parameters

Parameter	Description
'Dynamic Jitter Buffer Minimum Delay' `configure voip > media rtp-rtcp > jitter-buffer-minimum-delay` [DJBufMinDelay]	Global parameter defining the minimum delay (in msec) of the device's dynamic Jitter Buffer. You can also configure the feature per specific calls, using IP Profiles (IpProfile_ JitterBufMinDelay). For a detailed description of the parameter and for configuring the feature, see Configuring IP Profiles. **Note:** If the feature is configured for a specific profile, the settings of the global parameter is ignored for calls associated with the profile.
'Dynamic Jitter Buffer Optimization Factor' `configure voip > media rtp-rtcp > jitter-buffer-optimization-factor` [DJBufOptFactor]	Global parameter defining the Dynamic Jitter Buffer frame error/delay optimization factor. You can also configure the feature per specific calls, using IP Profiles (IpProfile_ JitterBufOptFactor) . For a detailed description of the parameter and for configuring the feature, see Configuring IP Profiles . **Note:** If the feature is configured for a specific profile, the settings of the global parameter is ignored for calls associated with the profile.
'RTP Redundancy Depth' `configure voip > media rtp-rtcp > RTP-redundancy-depth` [RTPRedundancyDepth]	Global parameter that enables the device to generate RFC 2198 redundant packets. You can also configure this feature per specific calls, using IP Profiles (IpProfile_RTPRedundancyDepth). For a detailed description of the parameter and for configuring this feature in the IP Profiles table, see Configuring IP Profiles. **Note:** If this feature is configured for a specific IP Profile, the settings of this global parameter is ignored for calls associated with the IP Profile.
'RFC 2198 Payload Type' `configure voip > media rtp-rtcp > RTP-redundancy-payload-type` [RFC2198PayloadType]	Defines the RTP redundancy packet payload type (according to RFC 2198). The valid value is 96 to 127. The default is 104. **Note:** The parameter is applicable only if the RTPRedundancyDepth parameter is set to 1.
'Packing Factor' [RTPPackingFactor]	N/A (controlled internally by the device according to the selected coder).

Parameter	Description
'RFC 2833 TX Payload Type' `configure voip > media rtp-rtcp > telephony-events-payload-type-tx` [RFC2833TxPayloadType]	Defines the Tx RFC 2833 DTMF relay dynamic payload type for outbound calls. The valid range is 96 to 127. The default is 96. **Note:** When RFC 2833 payload type negotiation is used (i.e., the parameter FirstTxDTMFOption is set to 4), this payload type is used for the received DTMF packets. If negotiation isn't used, this payload type is used for receive and for transmit.
'RFC 2833 RX Payload Type' `configure voip > media rtp-rtcp > telephony-events-payload-type-rx` [RFC2833RxPayloadType]	Defines the Rx RFC 2833 DTMF relay dynamic payload type for inbound calls. The valid range is 96 to 127. The default is 96. **Note:** When RFC 2833 payload type negotiation is used (i.e., the parameter FirstTxDTMFOption is set to 4), this payload type is used for the received DTMF packets. If negotiation isn't used, this payload type is used for receive and for transmit.
[EnableDetectRemoteMACChange]	Determines whether the device changes the RTP packets according to the MAC address of received RTP packets and according to Gratuitous Address Resolution Protocol (GARP) messages. ■ [0] = Nothing is changed. ■ [1] = If the device receives RTP packets with a different source MAC address (than the MAC address of the transmitted RTP packets), then it sends RTP packets to this MAC address and removes this IP entry from the device's ARP cache table. ■ [2] = (Default) The device uses the received GARP packets to change the MAC address of the transmitted RTP packets. ■ [3] = Options 1 and 2 are used. **Note:** ■ For the parameter to take effect, a device reset is required. ■ If the device is located in a network subnet which is connected to other gateways using a router that uses Virtual Router Redundancy

Parameter	Description
	Protocol (VRRP) for redundancy, then set the parameter to 0 or 2.
'Forward Invalid RTP Packets' [RTPFWInvalidPacketHandling]	Defines the device's handling of invalid RTP and RTCP packets. ■ [0] **Forward Packets** = Forwards the invalid packets as is. ■ [1] **Forward Packets and Issue Warnings** = (Default) Forwards the invalid packets and issues warnings (sent to the Syslog) to notify of the invalid packets. ■ [2] **Drop Packets and Issue Warnings** = Drops the invalid packets and issues warnings to notify of the invalid packets. **Note:** ■ The parameter is applicable only if the IPProfile_TranscodingMode parameter is configured to **RTP Forwarding**.
'Forward Unknown RTP Payload Types' [RtpFWNonConfiguredPTHandling]	Defines the device's handling of RTP packets that are received with non-configured (unknown) payload types. ■ [0] **Handle as Invalid Packet** = (Default) Handles the packet as an invalid packet, according to the RTPFWInvalidPacketHandling parameter. ■ [1] **Handle as Valid Packet** = Handles the packet as a valid packet. **Note:** ■ The parameter is applicable only if the IPProfile_TranscodingMode parameter is configured to **RTP Forwarding**.
'RTP Base UDP Port' `configure voip > media rtp-rtcp > base-udp-port` [BaseUDPport]	Global parameter that defines the lower boundary of the UDP port used for RTP, RTCP (RTP port + 1) and T.38 (RTP port + 2). For more information on configuring the UDP port range, see Configuring RTP Base UDP Port. The range of possible UDP ports is 6,000 to

Parameter	Description
	65,535. The default base UDP port is 6000. **Note:** For the parameter to take effect, a device reset is required.
`configure voip > media rtp-rtcp > udp-port-spacing` [UdpPortSpacing]	Defines the port spacing ("jumps") of local UDP ports allocated by the device to media channels (legs) within the configured port range. ■ [4] = (Default) The device allocates ports in "jumps" of 4 ports. ■ [5] = The device allocates ports in "jumps" of 5 ports. ■ [10] = The device allocates ports in "jumps" of 10 ports. **Note:** ■ A device reset is required for the parameter to take effect. ■ For more information on configuring the UDP port range, see Configuring RTP Base UDP Port.
'T.38 Fax Session' `configure voip > sip-definition settings > t38-sess-imm-strt` [T38FaxSessionImmediateStart]	Enables fax transmission of T.38 "no-signal" packets to the terminating fax machine. ■ [0] **Disable** (default) ■ [1] **Immediate Start on Fax** = Device activates T.38 fax relay upon receipt of a re-INVITE with T.38 only in the SDP. ■ [2] **Immediate Start on Fax & Voice** = Device activates T.38 fax relay upon receipt of a re-INVITE with T.38 and audio media in the SDP. The parameter is used for transmission from fax machines connected to the device and located inside a NAT. Generally, the firewall blocks T.38 (and other) packets received from the WAN, unless the device behind NAT sends at least one IP packet from the LAN to the WAN through the firewall. If the firewall blocks T.38 packets sent from the termination IP fax, the fax fails. To overcome this, the device sends No-Op (“no-

Parameter	Description
	signal") packets to open a pinhole in the NAT for the answering fax machine. The originating fax does not wait for an answer, but immediately starts sending T.38 packets to the terminating fax machine. **Note:** To enable No-Op packet transmission, use the [NoOpEnable] and [NoOpInterval] parameters.
`configure voip > sip-definition settings > t38-use-rtp-port` [T38UseRTPPort]	Defines the port (with relation to RTP port) for sending and receiving T.38 packets. ■ [0] = (Default) Use the RTP port +2 to send/receive T.38 packets. ■ [1] = Use the same port as the RTP port to send/receive T.38 packets. **Note:** ■ For the parameter to take effect, you must reset the device.
'T38 Fax Max Buffer' `configure voip > sip-definition settings > t38-fax-mx-buff` [T38FaxMaxBufferSize]	Defines the maximum size (in bytes) of the device's T.38 buffer. This value is included in the outgoing SDP when T.38 is used for fax relay over IP. The valid range is 500 to 3000. The default is 3,000.
No-Op Packets Parameters	
`no-operation-enable` [NoOpEnable]	Enables the device to send RTP or T.38 No-Op packets during RTP or T.38 silence periods. This mechanism ensures that the NAT binding remains open. ■ [0] = Disable (default) ■ [1] = Enable **Note:** You can also enable the feature per IP Profile, using the 'Generate No-Op Packets' IP Profile parameter.
[NoOpInterval]	Defines the interval (msec) between each RTP or T.38 No-Op packet sent by the device during the

Parameter	Description
	silence period (i.e., no RTP/T.38 traffic). The valid range is 20 to 600,000. The default is 1,000. **Note:** To enable No-Op packet transmission, use the [NoOpEnable] parameter.
`no-operation-interval` [RTPNoOpPayloadType]	Defines the payload type of No-Op packets. The valid range is 96 to 127 (for the range of Dynamic RTP Payload Type for all types of non hard-coded RTP Payload types, refer to RFC 3551). The default is 120. **Note:** When configuring the parameter, ensure that its settings don't cause collisions with other payload types.
RTP Control Protocol Extended Reports (RTCP XR) Parameters For more information on RTCP XR, see Configuring RTCP XR.	
'Enable RTCP XR' `configure voip > media rtp-rtcp > voice-quality-monitoring-enable` [VQMonEnable]	Enables voice quality monitoring and RTCP XR, according to RFC 3611. ■ [0] **Disable** (default) ■ [1] **Enable Fully** = Calculates voice quality metrics, uses them for QoE calculations, reports them to OVOC (if configured), and sends them to remote side using RTCP XR. ■ [2] **Enable Calculation Only** = Calculates voice quality metrics, uses them for QoE calculations, reports them to OVOC (if configured), but does not send them to remote side using RTCP XR.
'Minimum Gap Size' [VQMonGMin]	Defines the voice quality monitoring - minimum gap size (number of frames). The default is 16.
'Burst Threshold' [VQMonBurstHR]	Defines the voice quality monitoring - excessive burst alert threshold. The default is -1 (i.e., no alerts are issued).
'Delay Threshold' [VQMonDelayTHR]	Defines the voice quality monitoring - excessive delay alert threshold. The default is -1 (i.e., no alerts are issued).

Parameter	Description
'R-Value Delay Threshold' [VQMonEOCRValTHR]	Defines the voice quality monitoring - end of call low quality alert threshold. The default is -1 (i.e., no alerts are issued).
'Tx RTCP Packets Interval' `configure voip > media rtp-rtcp > rtcp-interval` [RTCPInterval]	Defines the time interval (in msec) between adjacent RTCP XR reports. This interval starts from call establishment. Thus, the device can send RTCP XR reports during the call, in addition to at the end of the call. If the duration of the call is shorter than this interval, RTCP XR is sent only at the end of the call. The valid value range is 0 to 65,535. The default is 5,000.
'Disable RTCP XR Interval Randomization' `configure voip > media rtp-rtcp > disable-RTCP-randomization` [DisableRTCPRandomize]	Determines whether RTCP report intervals are randomized or whether each report interval accords exactly to the parameter RTCPInterval. ■ [0] **Disable** = (Default) Randomize ■ [1] **Enable** = No Randomize
'Publication IP Group ID' `publication-ip-group-id` [PublicationIPGroupID]	Defines the IP Group to where the device sends RTCP XR reports. By default, no value is defined.
'SBC RTCP XR Report Mode' `configure voip > sip-definition settings > sbc-rtcpxr-report-mode` [SBCRtcpXrReportMode]	Enables the sending of RTCP XR reports of QoE metrics at the end of each call session (i.e., after a SIP BYE). The RTCP XR is sent in the SIP PUBLISH message. ■ [0] **Disable** (default) ■ [1] **End of Call**

Answer and Disconnect Supervision Parameters

The answer and disconnect supervision parameters are described in the table below.

Table 60-36:Answer and Disconnect Parameters

Parameter	Description
'Broken Connection Mode' `configure voip > sip-`	Global parameter that defines the device's handling of calls if RTP packets are not received

Parameter	Description
`definition settings > disc-broken-conn` [DisconnectOnBrokenConnection]	within a user-defined timeout, configured by the [BrokenConnectionEventTimeout] parameter. You can also configure this feature per specific calls, using IP Profiles (IpProfile_ DisconnectOnBrokenConnection). For a detailed description of the parameter and for configuring this feature in the IP Profiles table, see Configuring IP Profiles. **Note:** If this feature is configured for a specific IP Profile, the settings of this global parameter is ignored for calls associated with the IP Profile.
'Broken Connection Timeout' `configure voip > sip-definition settings > broken-connection-event-timeout` [BrokenConnectionEventTimeout]	Defines the timeout interval (in 100-msec units) after which a call is disconnected if RTP packets are not received during an established call (i.e., RTP flow suddenly stops during the call). The valid range is from 3 (i.e., 3 x 100 = 300 msec) to approx. 2684354 (i.e., 74.5 hours). The default is 100 msec. **Note:** ■ The parameter is applicable only if the [DisconnectOnBrokenConnection] parameter is configured to [1]. ■ Currently, the feature functions only if Silence Suppression is disabled.
`configure voip > sbc settings > no-rtp-detection-timeout` [NoRTPDetectionTimeout]	Defines the timeout interval (in msec) after which a call is disconnected if RTP packets are not received within the interval. The timer begins from call setup and if no packets are received when the timer expires, the device disconnects the call. The valid range is 0 to 50000. The default is 0, which means that this timeout feature is disabled and that the device does not disconnect the call due to RTP packets not being received. **Note:** ■ If a call is already established and there is RTP, if at any stage during the call RTP packets are not detected for a user-defined interval, configured by [BrokenConnectionEventTimeout], the device

Parameter	Description
	disconnects the call, or routes it to an alternative destination, configured by the [IpProfile_DisconnectOnBrokenConnection] parameter. ■ The parameter is not applicable to direct media calls (see Direct Media Calls on page 666).

SBC Parameters

The SBC parameters are described in the table below.

Table 60-37:SBC Parameters

Parameter	Description
SBC-specific Parameters	
`configure voip > application > enable-sbc` [EnableSBCApplication]	Enables the Session Border Control (SBC) application. ■ [0] = Disable ■ [1] = Enable (default) **Note:** ■ For the parameter to take effect, a device reset is required. ■ The parameter is enabled by default only if the License Key contains at least one of the SBC-related capacity features (e.g., "SBC-Signaling"); otherwise, the parameter is disabled.
SBC Parameters	
'Terminate Inbound OPTIONS' `configure voip > sbc settings > sbc-terminate-options` [SBCTerminateOptions]	Enables the device to terminate incoming in-dialog SIP OPTIONS messages or forward them to the outbound leg. ■ [0] **Disable** ■ [1] **Enable** (default)
'Unclassified Calls' `configure voip > sbc settings`	Determines whether incoming calls that cannot be classified (i.e. classification process fails) to a Source IP Group are

Parameter	Description
`> unclassified-calls` [AllowUnclassifiedCalls]	rejected or processed. ■ [0] **Reject** = (Default) Call is rejected if classification fails. ■ [1] **Allow** = If classification fails, the incoming packet is assigned to a source IP Group (and subsequently processed) as follows: ✓ The source SRD is determined according to the SIP Interface to where the SIP-initiating dialog request is sent. The source IP Group is set to the default IP Group associated with this SRD. ✓ If the source SRD is ID 0, then source IP Group ID 0 is chosen. In case of any other SRD, then the first IP Group associated with this SRD is chosen as the source IP Group or the call. If no IP Group is associated with this SRD, the call is rejected.
'SBC Max Call Duration' `configure voip > sbc settings > sbc-mx-call-duration` [SBCMaxCallDuration]	Defines the maximum duration (in minutes) per SBC call (global). If the duration is reached, the device terminates the call. The valid range is 0 to 35,791, where 0 is unlimited duration. The default is 0. **Note:** You can also configure this feature per specific calls, using IP Profiles (IpProfile_SBCMaxCallDuration). For a detailed description of the parameter and for configuring this feature in the IP Profiles table, see Configuring IP Profiles. If this feature is configured for a specific IP Profile, the settings of this global parameter is ignored for calls associated with the IP Profile.
'SBC No Answer Timeout' `configure voip > sbc settings > sbc-no-alert-timeout` [SBCAlertTimeout]	Defines the timeout (in seconds) for SIP INVITE messages sent by the device (outbound IP routing). The device starts the timeout when it sends

Parameter	Description
	the INVITE message and when (if) it receives the first SIP 18x response (e.g., 180 Ringing) from the called party. The timeout that is started when the INVITE message is sent, is **only** used if no 18x response is received. If the timeout expires and no additional SIP response (for example, 200 OK) was received during this interval, the device releases the call. The valid range is 0 to 3600 seconds. the default is 600.
`configure voip > sbc settings > num-of-subscribes` [NumOfSubscribes]	Defines the maximum number of concurrent SIP SUBSCRIBE sessions permitted on the device. The valid value is any value between 0 and the maximum supported SUBSCRIBE sessions. When set to -1, the device uses the default value. For more information, contact the sales representative of your purchased device. **Note:** For the parameter to take effect, a device reset is required.
`configure voip > sbc settings > sbc-dialog-subsc-route-mode` [SBCInDialogSubscribeRouteMode]	Enables the device to route in-dialog, refresh SIP SUBSCRIBE requests to the "working" (has connectivity) proxy. ■ [0] = (Default) Disable – the device sends in-dialog, refresh SUBSCRIBES according to the address in the Contact header of the 200 OK response received from the proxy to which the initial SUBSCRIBE was sent (as per the SIP standard). ■ [1] = Enable – the device routes in-dialog, refresh SUBSCRIBES to the "working" proxy (regardless of the Contact header). The "working" proxy (address) is determined by the device's keep-alive mechanism for the Proxy Set that was used to route the initial SUBSCRIBE.

Parameter	Description
	Note: For this feature to be functional, ensure the following: ■ Keep-alive mechanism is enabled for the Proxy Set ('Proxy Keep-Alive' parameter is set to any value other than **Disable**). ■ Load-balancing between proxies is disabled ('Proxy Load Balancing Method' parameter is set to **Disable**).
`configure voip > sbc settings > sbc-max-fwd-limit` [SBCMaxForwardsLimit]	Defines the Max-Forwards SIP header value. The Max-Forwards header is used to limit the number of servers (such as proxies) that can forward the SIP request. The Max-Forwards value indicates the remaining number of times this request message is allowed to be forwarded. This count is decremented by each server that forwards the request. The parameter affects the Max-Forwards header in the received message as follows: ■ If the received header's original value is 0, the message is not passed on and is rejected. ■ If the received header's original value is less than the parameter's value, the header's value is decremented before being sent on. ■ If the received header's original value is greater than the parameter's value, the header's value is replaced by the user-defined parameter's value. The valid value range is 1-70. The default is 10.
'Play Tone on Connect Failure Behavior' `play-tone-on-connect-failure-behavior` [PlayToneOnConnectFailureBehavior]	Defines if the device connects or disconnects the call if it can't play the specified tone to the call party. This parameter relates to the feature that is described in Playing Tone upon Call Connect on page 820.

Parameter	Description
	■ [0] **Disconnect** (Default) ■ [1] **Ignore**
`configure voip > sip-definition settings > force-generate-to-tag` [ForceGenerateToTag]	Enables the device to generate the 'tag' parameter's value in the SIP To header. This is applied to the first SIP response, received from the called party, which the device sends to the dialog-initiating SIP user agent (caller). In other words, this device-generated To tag overwrites the original To tag generated by the called party. All SIP messages between the device and caller use this generated To tag, while all SIP messages between the device and called party use the To tag generated by the called party. As the device-generated To tag value is short (up to 12 characters), this feature may be useful for SIP UAs that cannot handle long tag values. An example of the To tag: `To: Alice@company.com; tag = 977748484 9@10.10.1.110` ■ [0] = Disable (default). The device forwards the To tag transparently between the SIP UAs. ■ [1] = Enable. The device generates the To tag in the response sent to the initiator of the SIP dialog. **Note:** The feature is applicable only if the 'SBC Operation Mode' parameter is configured to **B2BUA**. This can be configured in the SRD and IP Groups table. However: ■ The IP Group's 'SBC Operation Mode' parameter takes precedence over the SRD's 'SBC Operation Mode' parameter. For example, if the IP Group is configured for B2BUA but its' associated SRD is not, then the tag-generation feature can function.

Parameter	Description
	■ If the IP Group's 'SBC Operation Mode' parameter is not configured (-1), the tag-generation feature for the IP Group is functional only if its' associated SRD is configured for B2BUA. ■ For call routing between IP Groups, the feature can only function if both IP Groups are configured for B2BUA, or if one or both of them is not configured (-1), but the associated SRD is configured for B2BUA.
'Session-Expires' `configure voip > sbc settings > sbc-sess-exp-time` [SBCSessionExpires]	Defines the SBC session refresh timer (in seconds) in the Session-Expires header of outgoing INVITE messages. The valid value range is 90 (according to RFC 4028) to 86400. The default is 180.
'Minimum Session-Expires' `configure voip > sbc settings > min-session-expires` [SBCMinSE]	Defines the minimum amount of time (in seconds) between session refresh requests in a dialog before the session is considered timed out. This value is conveyed in the SIP Min-SE header. The valid range is 0 (default) to 1,000,000, where 0 means that the device does not limit Session-Expires.
`configure voip > sbc settings > sbc-session-refresh-policy` [SBCSessionRefreshingPolicy]	Defines the SIP user agent responsible for periodically sending refresh requests for established sessions (active calls). The session refresh allows SIP UAs or proxies to determine the status of the SIP session. When a session expires, the session is considered terminated by the UAs, regardless of whether a SIP BYE was sent by one of the UAs. The SIP Session-Expires header conveys the lifetime of the session, which is sent in re-INVITE or UPDATE requests (session refresh requests). The 'refresher=' parameter in the Session-Expires header (sent in the initial INVITE or subsequent 2xx response)

Parameter	Description
	indicates who sends the session refresh requests. If the parameter contains the value 'uac', the device performs the refreshes; if the parameter contains the value 'uas', the remote proxy performs the refreshes. An example of the Session-Expires header is shown below: `Session-Expires: 4000;refresher=uac` Thus, the parameter is useful when a UA does not support session refresh requests or does not support the indication of who performs session refresh requests. In such a scenario, the device can be configured to perform the session refresh requests. ■ [0] = (Default) Remote Refresher. The UA (proxy) performs the session refresh requests. The device indicates this to the UA by sending the SIP message with the 'refresher=' parameter in the Session-Expires header set to 'uas'. ■ [1] = SBC Refresher. The device performs the session refresh requests. The device indicates this to the UA by sending the SIP message with the 'refresher=' parameter in the Session-Expires header set to 'uac'. **Note:** ■ The time values of the Session-Expires (session refresh interval) and Min-SE (minimum session refresh interval) headers can be configured using the [SBCSessionExpires] and [SBCMinSE] parameters, respectively.
'User Registration Grace Time' `configure voip > sbc settings > sbc-usr-reg-grace-time` [SBCUserRegistrationGraceTime]	Defines additional time (in seconds) to add to the registration expiry time of users that are registered with the device. The valid value is 0 to 15,500,000. The default is 0. For more information, see Registration

Parameter	Description
	Refreshes.
'DB Routing Search Mode' configure voip > sbc settings > sbc-db-route-mode [SBCDBRoutingSearchMode]	Defines the method for searching a registered user in the device's User Registration database when a SIP INVITE message is received for routing to or from a user. If the registered user is found (i.e., destination URI in INVITE), the device routes the call to the user's corresponding contact address specified in the database. ■ [0] **All permutations** = (Default) ✓ **To User:** Device searches for the user in the database using the entire Request-URI (user@host). If not found, it searches for the user part of the Request-URI. For example, it first searches for "4709@joe.company.com" and if not found, it searches for "4709". ✓ **From User:** Device searches for the user in the database using the entire From header AOR (user@host). If not found, it searches for the user part of the From header AOR. For example, it first searches for "4709@domain.com" and if not found, it searches for "4709". ■ [1] **Dest URI dependant** = ✓ **To User:** Device searches for the user in the database using the entire Request-URI (user@host) **only**. For example, it searches for "4709@joe.company.com". ✓ **From User:** Device searches for the user in the database using the entire From header AOR (user@host) **only**. For example, for “From: <sip:4709@domain.com>", the device searches for “4709@domain.com”.

Parameter	Description
	Note: If the Request-URI contains the "tel:" URI or "user=phone" parameter, the device searches only for the user part.
'Handle P-Asserted-Identity' `configure voip > sbc settings > p-assert-id` [SBCAssertIdentity]	Global parameter that defines the handling of the SIP P-Asserted-Identity header. You can also configure this feature per specific calls, using IP Profiles (IpProfile_SBCAssertIdentity). For a detailed description of the parameter and for configuring this feature in the IP Profiles table, see Configuring IP Profiles. **Note:** If this feature is configured for a specific IP Profile, the settings of this global parameter is ignored for calls associated with the IP Profile.
'Keep original user in Register' `configure voip > sbc settings > keep-contact-user-in-reg` [SBCKeepContactUserinRegister]	Defines the device's handling of the SIP Contact header in REGISTER requests which it forwards as the outgoing message. ■ [0] **Do not keep user; override with unique identifier** = (Default) The device replaces the user part of the Contact header with a unique value, for example: ✓ Incoming Contact Header: <sip:123@domain.com> ✓ Outgoing Contact Header: <sip:FEU1-7-1-3@SBC> ■ [1] **keep user without unique identifier** = The device retains the original user part value of the Contact header in the outgoing REGISTER request. ■ [2] **Keep user; add unique identifier as URI parameter** = The device retains the original user part value of the Contact header in the outgoing REGISTER request. In addition, it adds the special URI parameter "ac-feu=<identifier>" to the Contact header, which is used to differentiate between two SIP entities

Parameter	Description
	with the same user part. The identifier value is generated by the device. ✓ Incoming Contact Header: <sip:123@domain.com> ✓ Outgoing Contact Header: <sip:123@SBC;ac-feu=1-7-1-3> **Note:** ■ The parameter is applicable only to REGISTER messages received from User-type IP Groups which are sent to Server-type IP Groups. ■ Depending on the 'Remote Representation Mode' parameter of the IP Profiles table (IpProfile_ SBCRemoteRepresentationMode), the host part in the SIP Contact header can replaced by the device's IP address or by the value of the 'SIP Group Name' parameter (configured in the IP Groups table).
'URI Comparison Excluded Parameters' `config-voip > sbc settings > uri-comparison-excluded-params` [SBCURIComparisonExcludedParams]	Defines which URI parameters are excluded when the device compares the URIs of two incoming dialog-initiating SIP requests (e.g., INVITEs) to determine if they were sent from a user that is registered in the device's registration database (registered AOR and corresponding Contact URI), during Classification. The value of the parameter is a free-text string, which cannot be empty. You can configure it to any sequence of parameters, separated by commas (e.g., "transport, maddr, ttl"). Alternatively, you can configure it to one of the following values (case-insensitive): ■ **All** = (Default) All URI parameters (except the "gr" (gruu) parameter, "user=phone" parameter, and the AudioCodes proprietary "ac-int"

Parameter	Description

	parameter) and ports are excluded in the comparison of the two URIs. Therefore, if there are two different registrations of the same user whose Contacts are differentiated only by ports and/or a proprietary parameter, the device considers them to be the same single registration, even though they are different registrations.
	■ **None** = All URI parameters and ports are included in the comparison of the two URIs.
	■ **Port** = The ports of the URIs are excluded in the comparison of the two URIs, but all other URI parameters are included in the comparison. "port" can be combined with other URI parameters that you want excluded (e.g., port, transport, proprietary-param).
	For example, if two SIP requests are received with different Contact header values, as shown below (in bold) and the parameter is configured to **All**, then the device considers these requests as received from the same registered user as it disregards the port (5060 and 5070), 'transport', and 'tt'l parameters in its comparison. If configured to **None**, the device considers these requests as received from two different registered users.
	`Contact: <sip:1000@172.17.142.105:`**`5060;`**`transport=`**`tcp;`**`ttl=`**`10`**`>`
	`Contact: <sip:1000@172.17.142.105:`**`5070;`**`transport=`**`tls;`**`ttl=`**`20`**`>`
	Note: The AudioCodes proprietary "feu" string value for the user part must be included in the Contact header of REGISTER requests that the device forwards to the registrar server when the parameter is

Parameter	Description
	configured to a non-default value (i.e., not **All**). Therefore, if you configure the parameter to a non-default value, the SBCKeepContactUserInRegister parameter must not be configured to **Keep User Without Unique Identifier** (1).
'SBC REFER Behavior' `configure voip > sbc settings > sbc-refer-bhvr` [SBCReferBehavior]	Global parameter that defines the handling of SIP REFER requests. You can also configure this feature per specific calls, using IP Profiles (IpProfile_ SBCRemoteReferBehavior). For a detailed description of the parameter and for configuring this feature in the IP Profiles table, see Configuring IP Profiles. **Note:** If this feature is configured for a specific IP Profile, the settings of this global parameter is ignored for calls associated with the IP Profile.
`configure voip > sbc settings > sbc-xfer-prefix` [SBCXferPrefix]	When the SBCReferBehavior is set to 1, the device, while interworking the SIP REFER message, adds the prefix "T~&R-" to the user part of the URI in the Refer-To header. After this, the device can receive an INVITE with such a prefix (the INVITE is sent by the UA that receives the REFER message or 302 response). If the device receives an INVITE with such a prefix, it replaces the prefix with the value defined for the SBCXferPrefix parameter. By default, no value is defined. **Note:** This feature is also applicable to 3xx redirect responses. The device adds the prefix "T~&R-" to the URI user part in the Contact header if the SBC3xxBehavior parameter is set to 1.
`configure voip > sbc settings > sbc-3xx-bhvt` [SBC3xxBehavior]	Global parameter that defines the handling of SIP 3xx redirect responses. You can also configure this feature per specific calls, using IP Profiles (IpProfile_ SBCRemote3xxBehavior). For a detailed

Parameter	Description
	description of the parameter and for configuring this feature in the IP Profiles table, see Configuring IP Profiles. **Note:** If this feature is configured for a specific IP Profile, the settings of this global parameter is ignored for calls associated with the IP Profile.

`configure voip > sbc settings > enforce-media-order`
[SBCEnforceMediaOrder]
	Enables the device to include all previously negotiated media lines within the current session ('m=' line) in the SDP offer-answer exchange (RFC 3264).
	■ [0] = (Default) Disable
	■ [1] = Enable
	For example, assume a call (audio) has been established between two endpoints and one endpoint wants to subsequently send an image in the same call session. If the parameter is enabled, the endpoint includes the previously negotiated media type (i.e., audio) with the new negotiated media type (i.e., image) in its SDP offer:
	`v=0`
	`o=bob 2890844730 2890844731 IN IP4 host.example.com`
	`s=`
	`c=IN IP4 host.example.com`
	`t=0 0`
	`m=audio 0 RTP/AVP 0`
	`m=image 12345 udptl t38`
	If the parameter is disabled, the only 'm=' line included in the SDP is the newly negotiated media (i.e., image).

'SBC Diversion URI Type' `configure voip > sbc settings > sbc-diversion-uri-type` [SBCDiversionUriType]	Defines the URI type to use in the SIP Diversion header of the outgoing SIP message. ■ [0] **Transparent** = (Default) The device does not change the URI and leaves it as is. ■ [1] **Sip** = The "sip" URI is used.

Parameter	Description
	■ [2] **Tel** = The "tel" URI is used. **Note:** The parameter is applicable only if the Diversion header is used. The SBCDiversionMode and SBCHistoryInfoMode parameters in the IP Profiles table determine the call redirection (diversion) SIP header to use - History-Info or Diversion.
`configure voip > sbc settings > sbc-server-auth-mode` [SBCServerAuthMode]	Defines whether authentication of the SIP client is done locally (by the device) or by a RADIUS server. ■ [0] = (Default) Authentication is done by the device (locally). ■ [1] = Authentication is done by an RFC 5090 compliant RADIUS server. ■ [2] = Authentication is done according to the Draft Sterman-aaa-sip-01 method. **Note:** ■ Currently, option [1] is not supported. ■ The parameter is overridden by the IP Group parameter, 'SBC Server Authentication' (IPGroup_ TypeSBCServerAuthType).
'Lifetime of nonce' `configure voip > sbc settings > lifetime-of-nonce` [AuthNonceDuration]	Defines the lifetime (in seconds) that the current nonce is valid for server-based authentication. The device challenges a message that attempts to use a server nonce beyond this period. The parameter is used to provide replay protection (i.e., ensures that old communication streams are not used in replay attacks). The valid value range is 30 to 600. The default is 300.
'Authentication Challenge Method' `configure voip > sbc settings > auth-chlng-mthd` [AuthChallengeMethod]	Defines the type of server-based authentication challenge. ■ [0] **0** = (Default) Send SIP 401 "Unauthorized" with a WWW-

Parameter	Description
	Authenticate header as the authentication challenge response. ■ [1] **1** = Send SIP 407 "Proxy Authentication Required" with a Proxy-Authenticate header as the authentication challenge response.
'Authentication Quality of Protection' `configure voip > sbc settings > auth-qop` [AuthQOP]	Defines the authentication and integrity level of quality of protection (QoP) for digest authentication offered to the client. When the device challenges a SIP request (e.g., INVITE), it sends a SIP 401 response with the Proxy-Authenticate header or WWW-Authenticate header containing the 'qop' parameter. The QoP offered in the 401 response can be 'auth', 'auth-int', both 'auth' and 'auth-int', or the 'qop' parameter can be omitted from the 401 response. In response to the 401, the client needs to send the device another INVITE with the MD5 hash of the INVITE message and indicate the selected auth type. ■ [0] **0** = The device sends 'qop=auth' in the SIP response, requesting authentication (i.e., validates user by checking user name and password). This option does not authenticate the message body (i.e., SDP). ■ [1] **1** = The device sends 'qop=auth-int' in the SIP response, indicating required authentication and authentication with integrity (e.g., checksum). This option restricts the client to authenticating the entire SIP message, including the body, if present. ■ [2] **2** = (Default) The device sends 'qop=auth, auth-int' in the SIP response, indicating either authentication or integrity. This enables the client to choose 'auth' or 'auth-int'. If the client chooses 'auth-int', then the body is

Parameter	Description
	included in the authentication. If the client chooses 'auth', then the body is not authenticated. ■ [3] **3** = No 'qop' parameter is offered in the SIP 401 challenge message.
'SBC User Registration Time' `configure voip > sbc settings > sbc-usr-rgstr-time` [SBCUserRegistrationTime]	Global parameter that defines the duration (in seconds) of the periodic registrations that occur between the user and the device (the device responds with this value to the user). You can also configure this feature per specific calls, using IP Profiles (IpProfile_SBCUserRegistrationTime). For a detailed description of the parameter and for configuring this feature in the IP Profiles table, see Configuring IP Profiles. **Note:** If this feature is configured for a specific IP Profile, the settings of this global parameter is ignored for calls associated with the IP Profile.
'SBC Proxy Registration Time' `configure voip > sbc settings > sbc-prxy-rgstr-time` [SBCProxyRegistrationTime]	Defines the duration (in seconds) for which the user is registered in the proxy database (after the device forwards the REGISTER message). This value is sent in the Expires header. When set to 0, the device sends the Expires header's value as received from the user to the proxy. The valid range is 0 to 2,000,000 seconds. The default is 0.
`configure voip > sbc settings > sbc-rand-expire` [SBCRandomizeExpires]	Enables the device to change the expiry time in the Expires header of SIP 200 OK responses for user registration or subscription requests. The feature is useful in scenarios where multiple users may refresh their registration or subscription simultaneously, causing the device to handle many such sessions at a given time. This may result in an overload of the device (reaching maximum session capacity), preventing the establishment of

Parameter	Description
	new calls or preventing the handling of some user registration or subscription requests. However, when this feature is enabled, the device assigns a random expiry time to each user registration or subscription, ensuring future user registration and subscription requests are more distributed over time (i.e., do not all occur simultaneously). The valid value is 0 (disabled) to 20 (any value from 1 to 20 is considered enabled). The default is enabled (10). If disabled (i.e., 0), the device does not change the expiry time. If enabled, the device assigns a random expiry time as follows: ■ If the received expiry time is less than 610 sec, the device reduces the time by up to 10 sec. For example, if the received expiry time is 110 sec, the device reduces it to anywhere between 100 (i.e., 110 – 10) and 110 sec. ■ If the received expiry time is greater than 610 sec, the device reduces the time to anywhere between 600 sec and the received expiry time. For example, if the received expiry time is 700 sec, the device reduces it to anywhere between 600 and 700 sec. ■ Minimum expiry time: ✓ The minimum expiry time that the device can reduce REGISTER messages to is 30 sec and SUBSCRIBE messages to 120 sec. For example, if the received expiry time in a REGISTER message is 35 sec, the device reduces the time to any value between 30 and 35 sec (and not by 10 seconds -- between 25 and 35). ✓ If the received expiry time is less than the minimum (as stated

Parameter	Description
	above), the expiry time remains unchanged. For example, if the received expiry time in a REGISTER message is 18 sec, the device forwards the message with this same expiry time (i.e., 18). **Note:** ■ This feature does not apply to refresh REGISTER or SUBSCRIBE messages. ■ You can configure the device to change the received expiry time before forwarding it, using the SBCUserRegistrationTime parameter.
'SBC Survivability Registration Time' `configure voip > sbc settings > sbc-surv-rgstr-time` [SBCSurvivabilityRegistrationTime]	Defines the duration of the periodic registrations between the user and the device, when the device is in survivability state (i.e., when REGISTER requests cannot be forwarded to the proxy and are terminated by the device). When set to 0, the device uses the value set by the SBCUserRegistrationTime parameter for the device's response. The valid range is 0 to 2,000,000 seconds. The default is 0.
`configure voip > sbc settings > sas-notice` [SBCEnableSurvivabilityNotice]	Enables the device to notify Aastra IP phones that the device is currently operating in Survivability mode. ■ [0] = Disable ■ [1] = Enable For more information, see Enabling Survivability Display on Aastra IP Phones.
'SBC Dialog-Info Interworking' `configure voip > sbc settings > sbc-dialog-info-interwork` [EnableSBCDialogInfoInterworking]	Enables the interworking of dialog information (parsing of call identifiers in XML body) in SIP NOTIFY messages received from a remote application server. ■ [0] **Disable** (default) ■ [1] **Enable**

Parameter	Description
	For more information, see Interworking Dialog Information in SIP NOTIFY Messages.
`configure voip > sbc settings > sbc-keep-call-id` [SBCKeepOriginalCallId]	Global parameter that enables the device to use the same call identification (SIP Call-ID header value) received in incoming messages for the call identification in outgoing messages. The call identification value is contained in the SIP Call-ID header. You can also configure the feature per specific calls, using IP Profiles. For a detailed description of the parameter and for configuring the feature in the IP Profiles table, see Configuring IP Profiles.
`configure voip > sbc settings -> sbc-terminate-options` [SBCTerminateOptions]	Defines the handling of in-dialog SIP OPTIONS messages. ■ [0] = Disabled - the device forwards in-dialog SIP OPTIONS to the outbound peer. ■ [1] = (Default) Enabled - the device terminates in-dialog SIP OPTIONS and sends a 200 OK response to the peer that sent the OPTIONS message.
'Routing Timeout' `configure voip > sbc settings > sbc-routing-timeout` [SbcRoutingTimeout]	Defines the maximum duration (in seconds) that the device is prepared to wait for a response from external servers when a routing rule is configured to query an external server (e.g., LDAP server) on whose response the device uses to determine the routing destination. The valid value is 0 to 60. The default is 10. For more information, see Configuring a Routing Response Timeout on page 741.
'SBC GRUU Mode' `configure voip > sbc settings > sbc-gruu-mode` [SBCGruuMode]	Determines the Globally Routable User Agent (UA) URI (GRUU) support, according to RFC 5627. ■ [0] **None** = No GRUU is supplied to users. ■ [1] **As Proxy** = (Default) The device

Parameter	Description
	provides same GRUU types as the proxy provided the device's GRUU clients. ■ [2] **Temporary only** = Supply only temporary GRUU to users. (Currently not supported.) ■ [3] **Public only** = The device provides only public GRUU to users. ■ [4] **Both** = The device provides temporary and public GRUU to users. (Currently not supported.) The parameter allows the device to act as a GRUU server for its SIP UA clients, providing them with public GRUU's, according to RFC 5627. The public GRUU provided to the client is denoted in the SIP Contact header parameters, "pub-gruu". Public GRUU remains the same over registration expirations. On the other SBC leg communicating with the Proxy/Registrar, the device acts as a GRUU client. The device creates a GRUU value for each of its registered clients, which is mapped to the GRUU value received from the Proxy server. In other words, the created GRUU value is only used between the device and its clients (endpoints). `Public-GRUU: sip:userA@domain.com;gr=unique-id`
'BYE Authentication' `configure voip > sbc settings > sbc-bye-auth` [SBCEnableByeAuthentication]	Enables authenticating a SIP BYE request before disconnecting the call. This feature prevents, for example, a scenario in which the SBC SIP client receives a BYE request from a third-party imposer assuming the identity of a participant in the call and as a consequence, the call between the first and second parties is inappropriately disconnected. ■ [0] **Disable** (default)

Parameter	Description
	■ [1] **Enable** = The device forwards the SIP authentication response (for the BYE request) to the request sender and waits for the user to authenticate it. The call is disconnected only if the authenticating server responds with a 200 OK.
'SUBSCRIBE Trying' `configure voip > sbc settings > sbc-subs-try` [SBCSendTryingToSubscribe]	Enables the device to send a SIP 100 Trying response upon receipt of a SUBSCRIBE or NOTIFY message. ■ [0] **Disable** (default) ■ [1] **Enable**
`configure voip > sbc settings > sbc-100trying-upon-reinvite` [SBC100TryingUponReinvite]	Enables the device to send a SIP 100 Trying response upon receipt of a re-INVITE request. ■ [0] = (Default) Disable ■ [1] = Enable
'BroadWorks Survivability Feature' `configure voip > sbc settings > sbc-broadworks-survivability` [SBCExtensionsProvisioningMode]	Enables SBC user registration for interoperability with BroadSoft's BroadWorks server, to provide call survivability in case of connectivity failure with the BroadWorks server. ■ [0] **Disable** = (Default) Normal processing of REGISTER messages. ■ [1] **Enable** = Registration method for BroadWorks server. In a failure scenario with BroadWorks, the device acts as a backup SIP proxy server, maintaining call continuity between the enterprise LAN users (subscribers) and between the subscribers and the PSTN (if provided). **Note:** For a detailed description of this feature, see Enabling Auto-Provisioning of Subscriber-Specific Information of BroadWorks Server for Survivability.
'SBC Direct Media' `configure voip > sip-`	Enables the Direct Media feature (i.e., no Media Anchoring) for all SBC calls, whereby

Parameter	Description
`interface > sbc-direct-media` [SBCDirectMedia]	SIP signaling is handled by the device without handling the RTP/SRTP (media) flow between the user agents (UA). The RTP packets do not traverse the device. Instead, the two SIP UAs establish a direct RTP/SRTP flow between one another. Signaling continues to traverse the device with minimal intermediation and involvement to enable certain SBC abilities such as routing ■ [0] **Disable** = (Default) All calls traverse the device (i.e., no direct media). ■ [1] **Enable** = Direct media flow between endpoints for all SBC calls. **Note:** ■ The setting of direct media in the SIP Interfaces table overrides this global parameter. In other words, even if the parameter is disabled for direct media (i.e., Media Anchoring is enabled), if direct media is enabled for a SIP Interface (in the SIP Interfaces table), calls between endpoints belonging to the SIP Interface employ direct media. ■ For more information on No Media Anchoring, see Direct Media.
'Transcoding Mode' `configure voip > sbc settings > transcoding-mode` [TranscodingMode]	Global parameter that defines the voice transcoding mode (media negotiation). You can also configure this feature per specific calls, using IP Profiles (IpProfile_ TranscodingMode). For a detailed description of the parameter and for configuring this feature in the IP Profiles table, see Configuring IP Profiles. **Note:** If this feature is configured for a specific IP Profile, the settings of this global parameter is ignored for calls associated with the IP Profile.
'Preferences Mode' `configure voip > sbc settings`	Determines the order of the Extension coders (coders added if there are no

Parameter	Description
`> sbc-preferences` [SBCPreferencesMode]	common coders between SDP offered coders and Allowed coders) and Allowed coders (configured in the Allowed Audio Coders Groups table) in the outgoing SIP message (in the SDP). ■ [0] **Doesn't Include Extensions** = (Default) Extension coders are added at the end of the coder list. ■ [1] **Include Extensions** = Extension coders and Allowed coders are arranged according to their order of appearance in the Allowed Audio Coders Groups table. **Note:** ■ The parameter is applicable only if a Coders Group for Extension coders is assigned to the IP Profile (IPProfile_ SBCExtensionCodersGroupName).
'Reserve DSP on SDP Offer' `configure voip > sbc settings > reserve-dsp-on-sdp-offer` [ReserveDSPOnSDPOffer]	Enables the device to allocate DSP resources for a call at the SDP Offer or SDP Answer stage. ■ [0] **Disable** = The device allocates DSPs if available and required (e.g., for transcoding) for the call at the SDP Answer stage. ■ [1] **Enable** = (Default) The device allocates and reserves DSPs (if available) for the call at the SDP Offer. For more information on this feature, see Allocating DSPs on SDP Offer or Answer on page 674.
'SBC RTCP Mode' `configure voip > sbc settings > sbc-rtcp-mode` [SBCRTCPMode]	Global parameter that defines the handling of RTCP packets. You can also configure this feature per specific calls, using IP Profiles (IPProfile_SBCRTCPMode). For a detailed description of the parameter and for configuring this feature in the IP Profiles table, see Configuring IP Profiles.

Parameter	Description
	Note: If this feature is configured for a specific IP Profile, the settings of this global parameter is ignored for calls associated with the IP Profile.
`configure voip > sbc settings > sbc-send-invite-to-all-contacts` [SBCSendInviteToAllContacts]	Enables call forking of INVITE message received with a Request-URI of a specific contact registered in the device's database, to all users under the same AOR as the contact. ■ [0] **Disable** = (Default) Sends the INVITE only to the contact of the received Request-URI. ■ [1] **Enable** To configure call forking initiated by the device, see Initiating SIP Call Forking.
'SBC Shared Line Registration Mode' `configure voip > sbc settings > sbc-shared-line-reg-mode` [SBCSharedLineRegMode]	Enables the termination on the device of SIP REGISTER messages from secondary lines that belong to the Shared Line feature. ■ [0] **Disable** = (Default) Device forwards the REGISTER messages as is (i.e., not terminated on the device). ■ [1] **Enable** = REGISTER messages of secondary lines are terminated on the device. **Note:** The device always forwards REGISTER messages of the primary line.
'SBC Forking Handling Mode' `configure voip > sbc settings > sbc-forking-handling-mode` [SBCForkingHandlingMode]	Defines the handling of SIP 18x responses that are received due to call forking of an INVITE. ■ [0] **Latch On First** = (Default) Only the first 18x is forwarded to the INVITE-initiating UA. If SIP 18x with SDP is received, the device opens a voice stream according to the received SDP and disregards any subsequent 18x forking responses (with or without SDP). If the first response is 180 without SDP,

Parameter	Description
	the device sends it to the other side. ■ [1] **Sequential** = All 18x responses are forwarded, one at a time (sequentially) to the INVITE-initiating UA. If a 18x arrives with an offer only, then only the first offer is forwarded to the INVITE-initiating UA and subsequent 18x responses are discarded.
`configure voip > sbc settings > sbc-media-sync` [EnableSBCMediaSync]	Enables synchronization of media between two SIP user agents when a call is established between them. Media synchronization means that the media is properly negotiated (SDP offer/answer) between the user agents. In some scenarios, the call is established despite the media not being synchronized. This may occur, for example, in call transfer (SIP REFER) where the media between the transfer target and transferee are not synchronized. The device performs media synchronization by sending a re-INVITE immediately after the call is established in order for the user agents to negotiate the media (SDP offer/answer). ■ [0] = (Default) Disable. Media synchronization is performed only if the RTP mode (e.g., a=sendrecv, a=sendrecv, a=sendonly, a=recvonly, and a=inactive) between the user agents are different and synchronization is required. ■ [1] = Enable. Media synchronization is performed if the media, including RTP mode or any other media such as coders, is different and has not been negotiated between the user agents. ■ [2] = Never. Media synchronization is never performed.
'Remove SIPS from Non-Secured Transport' `configure voip > sbc settings > sbc-remove-sips-non-sec-`	Defines the SIP headers for which the device replaces “sips:” with “sip:” in the outgoing SIP-initiating dialog request (e.g., INVITE)

Parameter	Description
`transp` [SBCRemoveSIPSFromNonSecuredTransport]	when the destination transport type is unsecured (e.g., UDP). (The “sips:” URI scheme indicates secured transport, for example, TLS.) ■ [0] **Disable** = (Default) The device replaces “sips:” with “sip:” for the Request-URI and Contact headers only (and retains “sips:” for all other headers). ■ [1] **Enable** = The device replaces “sips:” with “sip:” for the Request-URI, Contact, From, To, P-Asserted, P-Preferred, and Route headers.
'SBC Fax Detection Timeout' `configure voip > sbc settings > sbc-fax-detection-timeout` [SBCFaxDetectionTimeout]	Defines the duration (in seconds) for which the device attempts to detect fax (CNG tone) immediately upon the establishment of a voice session. The interval starts from the establishment of the voice call. The valid value is 1 to any integer. The default is 10. The feature applies to faxes that are sent immediately after the voice channel is established (i.e., after 200 OK). You can configure the handling of fax negotiation by the device for specific calls, using IP Profiles configured in the IP Profiles table (see the IpProfile_ SBCRemoteRenegotiateOnFaxDetection parameter in Configuring IP Profiles).
'SIP Topology Hiding Mode' `configure voip > sbc settings > sip-topology-hiding-mode` [SIPTopologyHidingMode]	Enables the device to overwrite the host part in SIP headers with IP addresses, unless the relevant host name parameters of the IP Group ('SIP Group Name' and 'SIP Source Host Name') are configured: ■ Headers concerned with the source of the message are overwritten with the IP address of the IP Interface from where the device sends the message. ■ Headers concerned with the destination

Parameter	Description
	of the message are overwritten with the destination IP address. The parameter can be configured to one of the following values: ■ [0] **By Host Name Parameters Only** = (Default) The device overwrites the host part in the SIP headers according to the configuration of the IP Group's 'SIP Group Name' and 'SIP Source Host Name' parameters. If these parameters are empty, the device doesn't overwrite the host part of the headers. ■ [1] **Fallback to IP Addresses** = This option is applicable only to dialog-initiating requests and in-dialog REFER requests. ✓ If the 'SIP Group Name' parameter of the destination IP Group is empty, the device overwrites the host part of the following destination-related SIP headers with the destination IP address: Request-URI and P-Called-Party-ID for all types of requests, and To header for non-REGISTER requests. If the 'SIP Group Name' parameter is configured, the device overwrites the host part in these headers with the configured value. ✓ The source-related headers which are overwritten when the 'SIP Source Host Name' parameter is configured (From, P-Asserted-Identity, P-Preferred-Identity, Referred-By, P-Charge-Info, Remote-Party-ID, P-Associated-URI, Diversion, and History-info) are always overwritten. If the 'SIP Source Host Name' parameter of the destination IP Group is configured,

Parameter	Description
	the device overwrites the host part with the configured value. If the 'SIP Source Host Name' parameter of the destination IP Group is empty, the device overwrites the host part of the mentioned headers with the IP address of the device's IP Interface from where it sends the message. For more information on the IP Group parameters 'SIP Group Name' and 'SIP Source Host Name', see Configuring IP Groups on page 451.
Push Notification Service	
`configure voip > sbc settings > pns-reminder-period` [PNSReminderPeriod]	Defines the time (in seconds) before the user's registration with the device expires, at which the device sends an HTTP message to the Push Notification Server to trigger it into sending a push notification to the user to remind the user to send a REGISTER refresh message to the device. The valid value range is 30 to 300. The default is 120.
`configure voip > sbc settings > pns-register-timeout` [PNSRegisterTimeout]	Defines the maximum time (in seconds) that the device waits for a SIP REGISTER refresh message from the user, before it forwards an incoming SIP dialog-initiating request (e.g., INVITE) to the user. The valid value range is 5 to 50. The default is 30. When the device receives an incoming SIP dialog-initiating request whose destination is the user, it sends an HTTP message to the Push Notification Server to trigger it into sending the user a push notification so that the user can send a REGISTER refresh message to the device. If the device receives the REGISTER refresh message within this timeout, it forwards the incoming SIP request to the user. If the timeout expires

Parameter	Description
	and the device still hasn't received the REGISTER refresh message, the device rejects the call.

Supplementary Services

The SBC supplementary services parameters are described in the table below.

Table 60-38:SBC Supplementary Services Parameters

Parameter	Description
Emergency Call Preemption Parameters For more information on SBC emergency call preemption, Configuring Call Preemption for SBC Emergency Calls.	
'SBC Preemption Mode' `configure voip > sbc settings > sbc-preemption-mode` [SBCPreemptionMode]	Enables SBC emergency call preemption. ■ [0] **Disable** (default) ■ [1] **Enable**
'Emergency Message Condition' `configure voip > sbc settings > sbc-emerg-condition` [SBCEmergencyCondition]	Defines the index of the Message Condition rule in the Message Conditions table that is used to identify emergency calls. **Note:** The device applies the rule only after call classification (but before inbound manipulation).
'Emergency RTP DiffServ' `configure voip > sbc settings > sbc-emerg-rtp-diffserv` [SBCEmergencyRTPDiffServ]	Defines DiffServ bits sent in the RTP for SBC emergency calls. The valid value is 0 to 63. The default is 46.
'Emergency Signaling DiffServ' `configure voip > sbc settings > sbc-emerg-sig-diffserv` [SBCEmergencySignalingDiffServ]	Defines DiffServ bits sent in SIP signaling messages for SBC emergency calls. This is included in the SIP Resource-Priority header. The valid value is 0 to 63. The default is 40.

IP Media Parameters

The IP media parameters are described in the table below.

Table 60-39:IP Media Parameters

Parameter	Description
'IPMedia Detectors' `configure voip > media ipmedia > ipm-detectors-enable` [EnableDSPIPMDetectors]	Enables the device's DSP detectors for detection features such as AMD. ■ [0] **Disable** (default) ■ [1] **Enable** **Note:** ■ For the parameter to take effect, a device reset is required. ■ The DSP Detectors feature is available only if the device is installed with a License Key that includes this feature. For installing a License Key, see License Key.
'Number of Media Channels' `configure voip > sbc settings > media-channels` [MediaChannels]	Defines the maximum number of DSP channels that can be used for features requiring DSP resources, for example, coder transcoding, DTMF transcoding, and answer machine detection (AMD). This parameter is used to limit the use of DSP channels. The default is -1, meaning that the maximum number of DSP channels is according to the License Key ('DSP Channels'). For more information on the License Key, see License Key on page 871. **Note:** ■ For transcoding, make sure that the device's License Key includes a license for the number of DSP resources ('DSP Channels') and a license for the number of transcoding sessions ('Transcoding Sessions'). ■ Most SBC features that require DSP resources use two DSP channels. For example, if the device needs to perform coder transcoding between two endpoints where one uses the G.711 coder and the other G.729, and a maximum of 100 concurrent transcoding sessions need to be supported, the device uses 200 DSP channels. For this scenario, you would configure the parameter to 200.

Parameter	Description
	■ If you modify the parameter to a value that is less than the number of DSP channels currently being used by the device for currently active calls, the device allows these calls to continue (and does not terminate them).
Automatic Gain Control (AGC) Parameters	
'Enable AGC' `configure voip > media ipmedia > agc-enable` [EnableAGC]	Global parameter enabling the AGC feature. Enables the AGC mechanism. The AGC mechanism adjusts the level of the received signal to maintain a steady (configurable) volume level. ■ [0] **Disable** (default) ■ [1] **Enable** For a description of AGC, see Automatic Gain Control (AGC).
'AGC Slope' `configure voip > media ipmedia > agc-gain-slope` [AGCGainSlope]	Determines the AGC convergence rate: ■ [0] **0** = 0.25 dB/sec ■ [1] **1** = 0.50 dB/sec ■ [2] **2** = 0.75 dB/sec ■ [3] **3** = 1.00 dB/sec (default) ■ [4] **4** = 1.25 dB/sec ■ [5] **5** = 1.50 dB/sec ■ [6] **6** = 1.75 dB/sec ■ [7] **7** = 2.00 dB/sec ■ [8] **8** = 2.50 dB/sec ■ [9] **9** = 3.00 dB/sec ■ [10] **10** = 3.50 dB/sec ■ [11] **11** = 4.00 dB/sec ■ [12] **12** = 4.50 dB/sec ■ [13] **13** = 5.00 dB/sec ■ [14] **14** = 5.50 dB/sec

Parameter	Description
	■ [15] **15** = 6.00 dB/sec ■ [16] **16** = 7.00 dB/sec ■ [17] **17** = 8.00 dB/sec ■ [18] **18** = 9.00 dB/sec ■ [19] **19** = 10.00 dB/sec ■ [20] **20** = 11.00 dB/sec ■ [21] **21** = 12.00 dB/sec ■ [22] **22** = 13.00 dB/sec ■ [23] **23** = 14.00 dB/sec ■ [24] **24** = 15.00 dB/sec ■ [25] **25** = 20.00 dB/sec ■ [26] **26** = 25.00 dB/sec ■ [27] **27** = 30.00 dB/sec ■ [28] **28** = 35.00 dB/sec ■ [29] **29** = 40.00 dB/sec ■ [30] **30** = 50.00 dB/sec ■ [31] **31** = 70.00 dB/sec
'AGC Redirection' `configure voip > media ipmedia > agc-redirection` [AGCRedirection]	Determines the AGC direction. ■ [0] **0** = (Default) AGC works on signals from the TDM side. ■ [1] **1** = AGC works on signals from the IP side.
'AGC Target Energy' `configure voip > media ipmedia > agc-target-energy` [AGCTargetEnergy]	Defines the signal energy value (dBm) that the AGC attempts to attain. The valid range is 0 to -63 dBm. The default is -19 dBm.
'AGC Minimum Gain' `configure voip > media ipmedia > agc-min-gain` [AGCMinGain]	Defines the minimum gain (in dB) by the AGC when activated. The range is 0 to -31. The default is -20. **Note:** For the parameter to take effect, a device reset is required.

Parameter	Description
AGC Maximum Gain `configure voip > media ipmedia > agc-max-gain` [AGCMaxGain]	Defines the maximum gain (in dB) by the AGC when activated. The range is 0 to 18. The default is 15. **Note:** For the parameter to take effect, a device reset is required.
'AGC Disable Fast Adaptation' `configure voip > media ipmedia > agc-disable-fast-adaptation` [AGCDisableFastAdaptation]	Enables the AGC Fast Adaptation mode. ■ [0] **Disable** (default) ■ [1] **Enable** Note: For the parameter to take effect, a device reset is required.
Answering Machine Detector (AMD) Parameters For more information on AMD, see Answering Machine Detection (AMD).	
'Answer Machine Detector Sensitivity Parameter Suite' `configure voip > media ipmedia > amd-sensitivity-parameter-suit` [AMDSensitivityParameterSuit]	Global parameter that defines the AMD Parameter Suite to use. You can also configure this feature per specific calls, using IP Profiles (IpProfile_AMDSensitivityParameterSuit). For a detailed description of the parameter and for configuring this feature in the IP Profiles table, see Configuring IP Profiles. **Note:** If this feature is configured for a specific IP Profile, the settings of this global parameter is ignored for calls associated with the IP Profile.
'Answer Machine Detector Sensitivity Level' `configure voip > media ipmedia > amd-sensitivity-level` [AMDSensitivityLevel]	Global parameter that defines the AMD detection sensitivity level of the selected AMD Parameter Suite. You can also configure this feature per specific calls, using IP Profiles (IpProfile_AMDSensitivityLevel). For a detailed description of the parameter and for configuring this feature in the IP Profiles table, see Configuring IP Profiles. **Note:** If this feature is configured for a specific IP Profile, the settings of this global parameter is ignored for calls associated with the IP Profile.
'AMD Sensitivity File' [AMDSensitivityFileName]	Defines the name of the AMD Sensitivity file that contains the AMD Parameter Suites. **Note:**

Parameter	Description
	■ This file must be in binary format (.dat). You can use the DConvert utility to convert the original file format from XML to .dat. ■ You can load this file using the Web interface (see Loading Auxiliary Files).
[AMDSensitivityFileUrl]	Defines the URL path to the AMD Sensitivity file for downloading from a remote server for the Automatic Update mechanism.
[AMDMinimumVoiceLength]	Defines the AMD minimum voice activity detection duration (in 5-ms units). Voice activity duration below this threshold is ignored and considered as non-voice. The valid value range is 10 to 100. The default is 42 (i.e., 210 ms).
[AMDMaxGreetingTime]	Global parameter that defines the maximum duration that the device can take to detect a greeting message. You can also configure this feature per specific calls, using IP Profiles (IpProfile_AMDMaxGreetingTime). For a detailed description of the parameter and for configuring this feature in the IP Profiles table, see Configuring IP Profiles. **Note:** If this feature is configured for a specific IP Profile, the settings of this global parameter is ignored for calls associated with the IP Profile.
[AMDMaxPostGreetingSilenceTime]	Global parameter that defines the maximum duration of silence from after the greeting time is over, configured by [AMDMaxGreetingTime], until the device's AMD decision. You can also configure this feature per specific calls, using IP Profiles (IpProfile_ AMDMaxPostSilenceGreetingTime). For a detailed description of the parameter and for configuring this feature in the IP Profiles table, see Configuring IP Profiles. **Note:** If this feature is configured for a specific IP Profile, the settings of this global parameter is ignored for calls associated with the IP Profile.

Parameter	Description
`configure voip > gateway digital settings > amd-timeout` [AMDTimeout]	Defines the timeout (in msec) between receiving Connect messages from the Tel side and sending AMD results. The valid range is 1 to 30,000. The default is 2,000 (i.e., 2 seconds).
'AMD Beep Detection Mode' `configure voip > sip-definition settings > amd-beep-detection` [AMDBeepDetectionMode]	Defines the AMD beep detection mode. This mode detects the beeps played at the end of an answering machine message, by using the X-Detect header extension. The device sends a SIP INFO message containing the field values 'Type=AMD' and 'SubType=Beep'. This feature allows users of certain third-party, Application servers to leave a voice message after an answering machine plays the “beep”. ■ [0] **Disabled** (default) ■ [1] **Start After AMD** ■ [2] **Start Immediately**
'Answer Machine Detector Beep Detection Timeout' `configure voip > media ipmedia > amd-beep-detection-timeout` [AMDBeepDetectionTimeout]	Defines the AMD beep detection timeout (i.e., the duration that the beep detector functions from when detection is initiated). This is used for detecting beeps at the end of an answering machine message. The valid value is in units of 100 milliseconds, from 0 to 1638. The default is 200 (i.e., 20 seconds).
'Answer Machine Detector Beep Detection Sensitivity' `configure voip > media ipmedia > amd-beep-detection-sensitivity` [AMDBeepDetectionSensitivity]	Defines the AMD beep detection sensitivity for detecting beeps at the end of an answering machine message. The valid value is 0 to 3, where 0 (default) is the least sensitive.

Services

This section describes services-related parameters.

SIP-based Media Recording Parameters

The SIP-based media recording parameters are described in the table below.

Table 60-40:SIP-based Media Recording Parameters

Parameter	Description
'Recording Server (SRS) Destination Username' `configure voip > sip-definition sip-recording settings > siprec-server-dest-username` [SIPRecServerDestUsername]	Defines the SIP user part for the recording server. This user part is added in the SIP To header of the INVITE message that the device sends to the recording server. The valid value is a string of up to 50 characters. By default, no user part is defined.
'SIP Recording Metadata Format' `configure voip > sip-definition sip-recording settings > siprec-metadata-format` [SIPRecMetadataFormat]	Defines the format of the SIPRec recording metadata that the device generates in SIP messages sent to the SRS. ■ [0] **Legacy** = (Default) The device generates the recording metadata in a "legacy" format, whereby the user part of the participant URI (source or destination) is used as the ID. ■ [1] **RFC 7865** = The device generates the recording metadata in a format that is according to RFC 7865, whereby all IDs (e.g., participant ID) are in Base64 format. This metadata format also includes additional XML tags with association information (e.g., "<participantsessionassoc>").
'SIP Recording Time Stamp Format' `configure voip > sip-definition sip-recording settings > siprec-time-stamp` [SIPRecTimeStamp]	Defines the format of the device's time (timestamp) in SIP messages that are sent to the SIP Recording Server (SRS). ■ [0] **Local Time** = (Default) The device's local time (without the UTC time zone) is used for the timestamp. ■ [1] **UTC** = The device's UTC time is used for the timestamp. **Note:** The timestamp is contained in the XML body of the SIP message. If the timestamp uses the UTC time, the time is suffixed with the letter "Z", for example: `<associate-time>2017-09-07T06:33:38Z</associate-time>`
'Video Recording Sync Timeout' `configure voip > sip-`	Defines the video synchronization timeout (in msec), which is applicable when the device also records the

Parameter	Description
`definition sip-recording settings > video-rec-sync-timeout` [VideoRecordingSyncTimeout]	video stream of audio-video calls for SIPRec. If the SIP 200 OK from the SRS is not received within this timeout, the device connects the video stream between the UAs (instead of waiting for the 200 OK). The valid value is 100 to 5,000. The default is 2,000.

RADIUS and LDAP Parameters

This section describes the RADIUS and LDAP parameters.

General Parameters

The general RADIUS and LDAP parameters are described in the table below.

Table 60-41:General RADIUS and LDAP Parameters

Parameter	Description
'Use Local Users Database' `configure system > mgmt-auth > use-local-users-db` [MgmtUseLocalUsersDatabase]	Defines when the device uses the Local Users table and Authentication server (LDAP or RADIUS) for authenticating users (based on login username-password credentials) attempting to log in to the device's management interface (e.g., Web or CLI). ■ [0] **When No Auth Server Defined** = (Default) If the Authentication server denies user access, no "fallback" to the device's Local Users table occurs and the user is denied access. ■ [1] **Always** = If the Authentication server denies user access, the device uses the Local Users table to authenticate the user. **Note:** ■ If there is no response from the Authentication server (connection timeout), you can configure (using the MgmtBehaviorOnTimeout parameter) whether the device denies access or whether it uses the Local Users table to authenticate the user. ■ If you have not configured an Authentication server, the device uses the Local Users table to authenticate the user.
'Behavior upon Authentication	Defines the device's response when a connection

Parameter	Description
Server Timeout' `configure system > mgmt-auth > timeout-behavior` [MgmtBehaviorOnTimeout]	timeout occurs with the LDAP/RADIUS server. ■ [0] **Deny Access** = User is denied access to the management platform. ■ [1] **Verify Access Locally** = (Default) Device verifies the user's credentials in its Local Users table (local database). **Note:** The parameter is applicable to LDAP- and RADIUS-based management-user login authentication.
'Default Access Level' `configure system > mgmt-auth > default-access-level` [DefaultAccessLevel]	Defines the default access level for the device when the LDAP/RADIUS response doesn't include an access level attribute for determining the user's management access level. The valid range is 0 to 255. The default is 200 (i.e., Security Administrator). **Note:** ■ The parameter is applicable to LDAP- and RADIUS-based management-user login authentication and authorization. ■ If a user is not associated with any LDAP Group (at the LDAP server), the device automatically uses the value of this parameter as the access level.

RADIUS Parameters

The RADIUS parameters are described in the table below.

Table 60-42:RADIUS Parameters

Parameter	Description
General RADIUS Parameters	
'Enable RADIUS Access Control' `configure system > radius settings > enable` [EnableRADIUS]	Enables the RADIUS application. ■ [0] **Disable** (default) ■ [1] **Enable** **Note:** For the parameter to take effect, a device reset is required.

Parameter	Description
[RadiusTrafficType]	Defines the device's network interface for communicating (RADIUS traffic) with the RADIUS server (s). ■ [0] = (Default) OAMP ■ [1] = Control **Note:** If set to Control, only one Control interface must be configured in the IP Interfaces table; otherwise, RADIUS communication will fail.
'RADIUS VSA Vendor ID' `configure system > radius settings > vsa-vendor-id` [RadiusVSAVendorID]	Defines the vendor ID that the device accepts when parsing a RADIUS response packet. The valid range is 0 to 0xFFFFFFFF. The default is 5003.
[MaxRADIUSSessions]	Defines the number of concurrent calls that can communicate with the RADIUS server (optional). The valid range is 0 to 240. The default is 240.
'RADIUS Packets Retransmission' [RADIUSRetransmission]	Defines the number of RADIUS retransmission retries when no response is received from the RADIUS server. See also the RadiusTo parameter. The valid range is 1 to 10. The default is 1.
'RADIUS Response Time Out' [RadiusTO]	Defines the time interval (in seconds) that the device waits for a response before it performs a RADIUS retransmission. See also the RADIUSRetransmission parameter. The valid range is 1 to 30. The default is 2.
RADIUS Accounting Parameters	
'RADIUS Accounting Type' `configure voip > sip-definition settings > radius-accounting` [RADIUSAccountingType]	Defines at what stage of the call RADIUS accounting messages are sent to the RADIUS accounting server. ■ [0] **At Call Release** = (Default) Sent at call release only. ■ [1] **At Connect & Release** = Sent at call connect and release. ■ [2] **At Setup & Release** = Sent at call setup and release.

Parameter	Description
'AAA Indications' `configure system > cdr > aaa-indications` [AAAIndications]	Enables the Authentication, Authorization and Accounting (AAA) indications. ■ [0] **None** = (Default) No indications. ■ [3] **Accounting Only** = Only accounting indications are used.
RADIUS User Authentication Parameters	
'Use RADIUS for Web/Telnet Login' `configure system > radius settings > enable-mgmt-login` [WebRADIUSLogin]	Enables RADIUS queries for Web and Telnet login authentication. When enabled, logging into the device's Web and Telnet embedded servers is done through a RADIUS server. The device communicates with a user-defined RADIUS server and verifies the given username and password against a remote database in a secure manner. ■ [0] **Disable** (default) ■ [1] **Enable** **Note:** ■ For RADIUS login authentication to function, you must also configure the EnableRADIUS parameter to 1 (Enable). ■ RADIUS authentication requires HTTP basic authentication, where the username and password are transmitted in clear text over the network. Therefore, it's recommended to set the HTTPSOnly parameter to 1 to force the use of HTTPS, since the transport is encrypted.
'Password Local Cache Mode' `configure system > radius settings > local-cache-mode` [RadiusLocalCacheMode]	Defines the device's mode of operation regarding the timer (configured by the parameter RadiusLocalCacheTimeout) that determines the validity of the username and password (verified by the RADIUS server). ■ [0] **Absolute Expiry Timer** = When you access a Web page, the timeout doesn't reset, instead it continues decreasing. ■ [1] **Reset Timer Upon Access** = (Default) Upon each access to a Web page, the timeout always resets (reverts to the initial value configured by

Parameter	Description
	RadiusLocalCacheTimeout).
'Password Local Cache Timeout' `configure system > radius settings > local-cache-timeout` [RadiusLocalCacheTimeout]	Defines the time (in seconds) the locally stored username and password (verified by the RADIUS server) are valid. When this time expires, the username and password become invalid and a must be re-verified with the RADIUS server. The valid range is 1 to 0xFFFFFF. The default is 300 (5 minutes). ■ [-1] = Never expires. ■ [0] = Each request requires RADIUS authentication.
'RADIUS VSA Access Level Attribute' `configure system > radius settings > vsa-access-level` [RadiusVSAAccessAttribute]	Defines the code that indicates the access level attribute in the Vendor Specific Attributes (VSA) section of the received RADIUS packet. The valid range is 0 to 255. The default is 35.

LDAP Parameters

The Lightweight Directory Access Protocol (LDAP) parameters are described in the table below.

Table 60-43:LDAP Parameters

Parameter	Description
'LDAP Service' `configure system > ldap settings > ldap-service` [LDAPServiceEnable]	Enables the LDAP feature. ■ [0] **Disable** (default) ■ [1] **Enable** **Note:** For the parameter to take effect, a device reset is required.
'LDAP Authentication Filter' `configure system > ldap settings > auth-filter` [LDAPAuthFilter]	Defines the LDAP search filter attribute for searching the login username in the directory's subtree for LDAP-based user authentication and authorization. You can use the dollar ($) sign to represent the username. For example, if the parameter is set to "(sAMAccountName=$)" and the user logs in with the username "SueM", the LDAP query is

Parameter	Description
	run for sAMAccountName=SueM.
'Use LDAP for Web > Telnet Login' `configure system > ldap settings > enable-mgmt-login` [MgmtLDAPLogin]	Enables LDAP-based management-user login authentication and authorization. ■ [0] **Disable** (default) ■ [1] **Enable** **Note:** For the parameter to take effect, a device reset is required.
[LDAPDebugMode]	Determines whether to enable the LDAP task debug messages. This is used for providing debug information regarding LDAP tasks. The valid value range is 0 to 3. The default is 0.
'LDAP Numeric Attribute' `ldap-numeric-attr` [LDAPNumericAttributes]	Defines up to five LDAP Attributes (separated by commas) for which the device employs LDAP query searches in the AD for numbers that may have characters between the digits. For more information, see Enabling LDAP Searches for Numbers with Characters.
'LDAP OCS Number Attribute Name' `configure voip > sip-definition settings > ldap-ocs-nm-attr` [MSLDAPOCSNumAttributeName]	Defines the name of the attribute that represents the user's Skype for Business number in the Microsoft AD database. The valid value is a string of up to 49 characters. The default is "msRTCSIP-Line".
LDAP PBX Number Attribute Name `configure voip > sip-definition settings > ldap-pbx-nm-attr` [MSLDAPPBXNumAttributeName]	Defines the name of the attribute that represents the user PBX number in the Microsoft AD database. The valid value is a string of up to 49 characters. The default is "telephoneNumber".
LDAP MOBILE Number Attribute Name `configure voip > sip-definition settings > ldap-mobile-nm-attr` [MSLDAPMobileNumAttributeName]	Defines the name of the attribute that represents the user Mobile number in the Microsoft AD database. The valid value is a string of up to 49 characters. The default is "mobile".
LDAP PRIVATE Number Attribute Name `configure voip > sip-`	Defines the name of the attribute that represents the user's private number in the

Parameter	Description
`definition settings > ldap-private-nm-attr` [MSLDAPPrivateNumAttributeName]	AD. If this value equals the value of the MSLDAPPrimaryKey or MSLDAPSecondaryKey parameter, then the device queries the AD for the destination number in this private attribute name; otherwise, the parameter is not used as a search key. The default is "msRTCSIP-PrivateLine".
'LDAP DISPLAY Name Attribute Name' `configure voip > sip-definition settings > ldap-display-nm-attr` [MSLDAPDisplayNameAttributeName]	Defines the attribute name that represents the Calling Name in the AD for LDAP queries based on calling number. The valid value is a string of up to 49 characters. The default is "displayName".
LDAP Primary Key `configure voip > sip-definition settings > ldap-primary-key` [MSLDAPPrimaryKey]	Defines the name of the attribute used as a query search key for the destination number in the AD. This is used instead of the "PBX" attribute name (configured by the MSLDAPPBXNumAttributeName parameter). The default is not configured.
LDAP Secondary Key `configure voip > sip-definition settings > ldap-secondary-key` [MSLDAPSecondaryKey]	Defines the name of the attribute used as the second query search key for the destination number in the AD, if the primary search key or PBX search is not found.
'LDAP Cache Service' `configure system > ldap settings > ldap-cache-enable` [LDAPCacheEnable]	Enables the LDAP cache service. ■ [0] **Disable** (default) ■ [1] **Enable** **Note:** ■ For the parameter to take effect, a device reset is required. ■ For more information on LDAP caching, see Configuring the Device's LDAP Cache.

HTTP-based Services

The HTTP-based service parameters are described in the table below.

Table 60-44:HTTP-based Service Parameters

Parameter	Description
'Quality Status' `configure system > http-services > routing-qos-status` [RoutingServerQualityStatus]	Enables QoS-based routing by the routing server. The device collects QoS metrics (media and signaling) and sends them to the routing server. ■ [0] **Disable** (default) ■ [1] **Enable** For more information, see Configuring QoS-Based Routing by Routing Server.
'Quality Status Rate' `configure system > http-services > routing-qos-status-rate` [RoutingServerQualityStatusRate]	Defines the rate (in sec) at which the device sends QoS reports to the routing server. The valid range is 15-3600. The default is 60. For more information, see Configuring QoS-Based Routing by Routing Server.
'Topology Status' `configure system > http-services > routing-server-group-status` [RoutingServerGroupStatus]	Enables the reporting of the device's topology status (using the REST TopologyStatus API command) to remote HTTP hosts. ■ [0] **Disable** (default) ■ [1] **Enable** For more information, see Configuring Remote Web Services on page 316
'Routing Server Registration Status' `configure system > http-services > routing-server-registration-status` [RoutingServerRegistrationStatus]	Enables the synchronization of the device's registration database (using the REST registrationStatus API command) with remote HTTP hosts. ■ [0] **Disable** (default)

Parameter	Description
	■ [1] **Enable** For more information, see Configuring Remote Web Services on page 316
Remote Monitoring For more information, see Remote Monitoring of Device behind NAT on page 1056.	
'Remote Monitoring' `configure system > http-services > remote-monitoring` [RemoteMonitoringEnable]	Enables the device to send monitoring reports to a remote monitoring server when the device is located behind NAT. ■ [0] = Disable (default) ■ [1] = Enable
'Reporting Period' `configure system > http-services > remote-monitor-reporting-period` [RemoteMonitoringPeriod]	Defines the time interval (in seconds) between each remote monitoring report that is sent to the monitoring server. The valid value is 30 to 65535. The default is 300.
'Device Status' `configure system > http-services > remote-monitor-status` [RemoteMonitoringDeviceEnable]	Enables the device to send a remote monitoring report of its status to the monitoring server. ■ [0] = Disable (default) ■ [1] = Enable
'Active Alarms' `configure system > http-services > remote-monitor-alarms` [RemoteMonitoringAlarmsEnable]	Enables the device to send a remote monitoring report of currently active alarms to the monitoring server. ■ [0] = Disable (default) ■ [1] = Enable
'Performance Indicators' `configure system > http-services > remote-monitor-kpi` [RemoteMonitoringPMEnable]	Enables the device to send a remote monitoring report of performance monitoring statistics to the monitoring server.

Parameter	Description
	■ [0] = Disable (default) ■ [1] = Enable
'Registration Status' `configure system > http-services > remote-monitor-registration` [RemoteMonitoringSIPUsersEnable]	Enables the device to send a remote monitoring report of users registered with the device to the monitoring server. ■ [0] = Disable (default) ■ [1] = Enable
Automatic Provisioning (see Configuring Web Service for Automatic Provisioning on page 338)	
'Enabled' check box `configure system > provision > enable` [ProvisionEnable]	Enables the provisioning feature. ■ [0] = (Default) Disables automatic provisioning. ■ [1] = Enables automatic provisioning. **Note:** For the parameter to take effect, a device reset is required.
'Retry Interval' `configure system > provision > retry-interval` [ProvisionRetryInterval]	Defines the time (in seconds) between each sent HTTP request that failed. The valid value is 10 to 360. The default is 30. **Note:** For the parameter to take effect, a device reset is required.
'Max Retries' `configure system > provision > max-retries` [ProvisionMaxRetries]	Defines the maximum number of attempts to send the request before provisioning is considered a failure. The valid value is 1 to 10. The default is 3. **Note:** For the parameter to take effect, a device reset is required.

Parameter	Description
'Server URL' `configure system > provision > server-url` [ProvisionServerURL]	Defines the provisioning server's URL path where the requests must be sent. The valid value is a string. By default, no value is defined. **Note:** For the parameter to take effect, a device reset is required.
'Server Username' `configure system > provision > server-username` [ProvisionServerUsername]	Defines the username for authentication with the server. The valid value is a string. By default, no value is defined. **Note:** For the parameter to take effect, a device reset is required.
'Server Password' `configure system > provision > server-password` [ProvisionServerPassword]	Defines the password for authentication with the server. The valid value is a string. By default, no value is defined. **Note:** For the parameter to take effect, a device reset is required.

HTTP Proxy Parameters

The HTTP Proxy service parameters are described in the table below.

Table 60-45:HTTP Proxy Service Parameters

Parameter	Description
'HTTP Proxy Application' `configure network > http-proxy > http-proxy-app` [HTTPProxyApplication]	Enables the HTTP Proxy application. ■ [0] **Disable** (default) ■ [1] **Enable** **Note:** For the parameter to take effect, a device reset is required.
'HTTP Proxy Debug Level' `configure network > http-proxy > http-proxy-debug-level`	Enables debugging of HTTP services and filtering of messages sent to the Syslog server by severity (debug)

Parameter	Description
[HTTPProxySyslogDebugLevel]	level. ■ [0] **No Debug** = (Default) Disable. ■ [1] **Info** = Enables debug and sends basic information in Syslog. The logged information includes details of access requests (HTTP requests to read or write files). ■ [2] **Notice** = Enables debug and sends normal, but significant information in Syslog. The logged information includes details of access requests (HTTP requests to read or write files). ■ [3] **Warning** = Enables debug and sends warning conditions in Syslog. ■ [4] **Error** = Enables debug and sends error conditions in Syslog. ■ [5] **Critical** = Enables debug and sends critical conditions in Syslog. ■ [6] **Alert** = Enables debug and sends conditions in Syslog that require immediate action. ■ [7] **Emergency** = Enables debug and sends conditions indicating unstable system in Syslog. **Note:** The NGINX Directive for this parameter is "error_log, access_log".
'Primary DNS Server IP' `dns-primary-server` [HTTPPrimaryDNS]	Defines the primary DNS server (in dotted-decimal notation), which is used for translating domain names into IP addresses for the HTTP service. By default, no IP address is defined.
'Secondary DNS Server IP' `dns-secondary-server` [HTTPSecondaryDNS]	Defines the secondary DNS server (in dotted-decimal notation), which is used for translating domain names

Parameter	Description
	into IP addresses for the HTTP service. By default, no IP address is defined.
'HTTP Proxy Global Address' `configure network > http-proxy > http-proxy-global-address` [HttpProxyGlobalAddress]	Defines the device's public IP address for the HTTP Proxy service, when the device is located behind NAT. The valid value is an IP address in dotted-decimal notation. The default is 0.0.0.0. For more information, see Configuring a Public IP Address for NGINX NAT Traversal on page 369.

61 Capacity for Signaling, Media and User Registrations

For supported capacity (SIP signaling, media and user registrations), refer to the device's *Release Notes*, which can be downloaded from AudioCodes website.

62 Technical Specifications

For technical specifications, click here to download the device's datasheet from the website.

This page is intentionally left blank.

International Headquarters

1 Hayarden Street,

Airport City

Lod 7019900, Israel

Tel: +972-3-976-4000

Fax: +972-3-976-4040

AudioCodes Inc.

80 Kingsbridge Rd

Piscataway, NJ 08854, USA

Tel: +1-732-469-0880

Fax: +1-732-469-2298

Contact us: https://www.audiocodes.com/corporate/offices-worldwide

Website: https://www.audiocodes.com/

Documentation Feedback: https://online.audiocodes.com/documentation-feedback

Document #: LTRT-42343