Enterprise Session Border Controllers

Mediant™ VoIP Media Gateways

MediaPack™ VoIP Media Gateways

IPmedia™ Media Servers

SIP CPE Product Reference Manual

SAS
Stand Alone Survivability
Continuous VoIP Service

MSBG
AudioCodes
Multi-Service
Business Gateway

SBC

HD VoIP
Sounds Better

Version 6.4

April 2012

Document # LTRT-52309

AudioCodes

Table of Contents

Reader's Notes

Notice

This document provides a reference guide for AudioCodes SIP-based Voice over IP (VoIP) devices.

Information contained in this document is believed to be accurate and reliable at the time of printing. However, due to ongoing product improvements and revisions, AudioCodes cannot guarantee accuracy of printed material after the Date Published nor can it accept responsibility for errors or omissions.

Some features mentioned in this document may not be supported for every product in this software version. You must consult the Release Notes for this version to verify whether your product is supported and/or if specific features are supported for your product. In cases where there are discrepancies between this Manual and the Release Notes, the information in the Release Notes supersedes that in this Manual.

Updates to this document and other documents can be viewed by registered customers at http://www.audiocodes.com/support.

This document is subject to change without notice.

Date Published: April-10-2012

Trademarks

AudioCodes, AC, AudioCoded, Ardito, CTI2, CTI², CTI Squared, HD VoIP, HD VoIP Sounds Better, InTouch, IPmedia, Mediant, MediaPack, NetCoder, Netrake, Nuera, Open Solutions Network, OSN, Stretto, TrunkPack, VMAS, VoicePacketizer, VoIPerfect, VoIPerfectHD, What's Inside Matters, Your Gateway To VoIP and 3GX are trademarks or registered trademarks of AudioCodes Limited. All other products or trademarks are property of their respective owners.

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 service are generally provided by AudioCodes' Distributors, Partners, and Resellers from whom the product was purchased. For technical support for products purchased directly from AudioCodes, or for customers subscribed to AudioCodes Customer Technical Support (ACTS), contact support@audiocodes.com.

Abbreviations and Terminology

Each abbreviation, unless widely used, is spelled out in full when first used.

Related Documentation

Manual Name
SIP CPE Release Notes
Mediant 600 Hardware Installation Manual
Mediant 1000 Hardware Installation Manual
Mediant 600 & Mediant 1000 SIP User's Manual
Mediant 800 MSBG Hardware Installation Manual
Mediant 800 MSBG User’s Manual
Mediant 800 Gateway & E-SBC Hardware Installation Manual
Mediant 800 Gateway & E-SBC User’s Manual
Mediant 1000 MSBG Hardware Installation Manual
Mediant 1000 MSBG User’s Manual
Mediant 1000 Gateway & E-SBC Hardware Installation Manual
Mediant 1000 Gateway & E-SBC User’s Manual
MSBG Series Data CLI Reference Guide
MSBG Series System & VoIP CLI Reference Guide
Mediant 2000 Hardware Installation Manual
Mediant 2000 SIP User's Manual
Mediant 3000 Hardware Installation Manual
Mediant 3000 SIP User's Manual
Mediant 4000 E-SBC Hardware Installation Manual
Mediant 4000 E-SBC User's Manual
CPE SIP Configuration Guide for IP Voice Mail

1 Introduction

This manual provides you with supplementary information on AudioCodes SIP-based, Voice-over-IP (VoIP) devices. This information is complementary to the information provided by the device's *User's Manual* and includes, for example, detailed descriptions on various supported features, proprietary applications, advanced configuration methods, and so on.

Notes:

- For AudioCodes devices supported in Release 6.4, refer to the *SIP CPE Release Notes*.
- For a detailed description on configuring the device, refer to the device's *User's Manual*.

Throughout this manual, unless otherwise specified, the following terms are used to refer to the devices listed in the previous section to indicate feature applicability:

Table 1-1: Devices Naming Convention

Term	Devices
Device	All products
MediaPack	MP-112, MP-114, MP-118, and MP-124
MSBG	Mediant 800 MSBG and Mediant 1000 MSBG
Analog	Analog interfaces for MediaPack, Mediant 600, Mediant 800 MSBG, Mediant 800 Gateway & E-SBC, Mediant 1000, Mediant 1000 MSBG, and Mediant 1000 Gateway & E-SBC
2000 Series	Mediant 2000 and IPmedia 2000
3000 Series	Mediant 3000 and IPmedia 3000
6310 Blade Series	TP-6310 and IPM-6310 blades
8410 Blade Series	TP-8410 and IPM-8410 blades
Digital PSTN	Digital PSTN interfaces for Mediant 600, Mediant 800 MSBG, Mediant 800 Gateway & E-SBC, Mediant 1000, Mediant 1000 MSBG, Mediant 1000 Gateway & E-SBC, Mediant 2000, and Mediant 3000
IPmedia Series	IPmedia 2000 and IPmedia 3000

Reader's Notes

2 Device Initialization

Note: This section is not applicable to MSBG.

This section describes the device's initialization process, including the different methods for initial configuration.

2.1 Startup Process

The startup process (illustrated in the following figure) begins when the device is reset. The device resets either by a manual (physical) reset, using the Web interface, using SNMP, or when there is a device irregularity. The startup process ends when the operational software is running. In the startup process, the device obtains its IP address, and software and configuration files.

After the device powers up or after it's physically reset, it broadcasts a BootRequest message to the network. If it receives a reply (from a BootP server), it changes its network parameters (IP address, subnet mask and default gateway address) to the values provided. If there is no reply from a BootP server and if DHCP is enabled (DHCPEnable = 1), the device initiates a standard DHCP procedure to configure its network parameters.

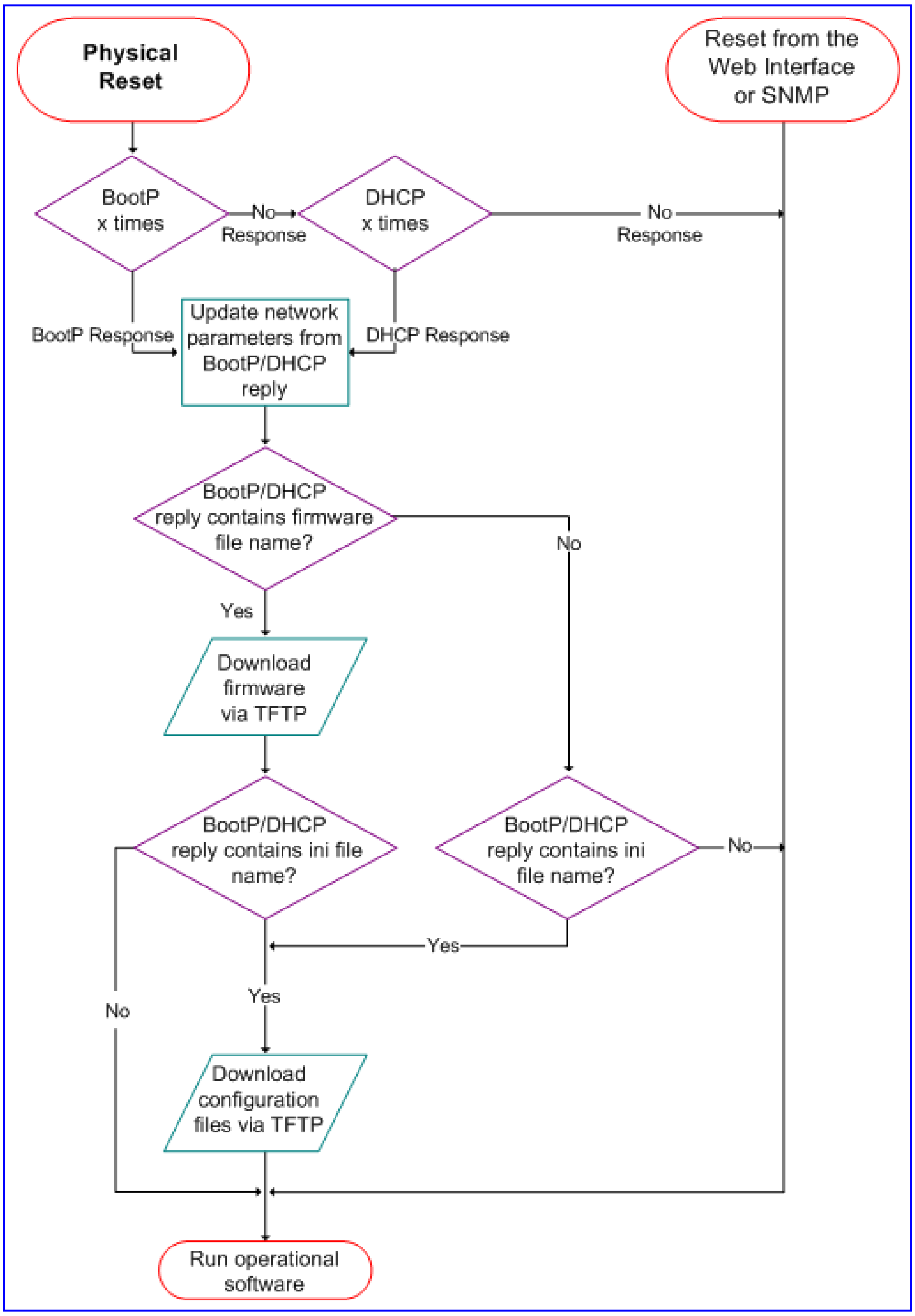
After changing the network parameters, the device attempts to load the device's operational firmware (*cmp*) and various configuration files from the TFTP server's IP address, received from the BootP/DHCP servers. If a TFTP server's IP address isn't received, the device attempts to load the *cmp* file and / or configuration files from a preconfigured TFTP server (see "Automatic Update Mechanism" on page 21). Thus, the device can obtain its network parameters from BootP or DHCP servers, and its software and configuration files from a different TFTP server (preconfigured in the *ini* configuration file).

If BootP/DHCP servers are not located or when the device is reset using the Web interface or SNMP, it retains its network parameters and attempts to load the *cmp* file and / or configuration files from a preconfigured TFTP server. If a preconfigured TFTP server doesn't exist, the device operates using the existing software and configuration files in its non-volatile memory.

Notes:

- After the operational software runs and if DHCP is configured, the device attempts to renew its lease with the DHCP server.
- Though DHCP and BootP servers are very similar in operation, the DHCP server includes some differences that could prevent its operation with BootP clients. However, many DHCP servers such as Windows™ NT DHCP server are backward compatible with BootP protocol and can be used for device configuration.
- By default, the duration between BootP/DHCP requests is one second (configured by the BootPDelay *ini* file parameter). By default, the number of requests is three (configured by the BootPRetries *ini* file parameter). Both parameters can also be set using the BootP command line switches (see Section "Using Command Line Switches" on page 193).

Figure 2-1: Startup Process


Physical Reset
Reset from the Web Interface or SNMP
BootP x times
No Response
DHCP x times
No Response
BootP Response
Update network parameters from BootP/DHCP reply
DHCP Response
BootP/DHCP reply contains firmware file name?
No
Yes
Download firmware via TFTP
BootP/DHCP reply contains ini file name?
BootP/DHCP reply contains ini file name?
No
Yes
Yes
No
Download configuration files via TFTP
Run operational software

2.2 Device Firmware

The device runs two distinct software programs:

- **Boot firmware:** Boot-loader firmware (also known as flash software), which resides on the device's non-volatile memory. When the device is reset, Boot firmware is initialized and the operational software is loaded from a TFTP server or integral non-volatile memory. Boot firmware is also responsible for obtaining the device's IP parameters and *ini* file name (used to obtain the device's configuration parameters) using integral BootP or DHCP clients. The Boot firmware version can be viewed in the Web interface (refer to the 'Device Information' page in the device's *User's Manual*). The last step the Boot firmware performs is to invoke the operational firmware.
- **Operational firmware file:** The operational firmware in the form of a cmp file (the software image file). This cmp file contains the device's main software, providing all the features described in this manual. The cmp file is usually burnt on the device's non-volatile memory so that it does not need to be externally loaded each time the device is reset.

2.3 Configuration Parameters and Files

The device's configuration is located in two types of files:

- **Initialization file:** An initialization (ini) text file containing the device's configuration parameters (referred to as *ini* file parameters and *ini* file table parameters). This file carries the file name extension *.ini.
- **Auxiliary files:** Contains the raw data used for various tasks such as Call Progress Tones. These files carry the file name extension *.dat.

These files are stored in the device's non-volatile memory (i.e., flash) and are set to factory defaults when shipped to the customer. The device starts up initially with this default configuration. Subsequently, these files can be modified and reloaded (to flash memory) using any of the following methods:

- BootP/TFTP during startup process (see "Using BootP / DHCP" on page 16).
- Device's Embedded Web server (refer to the device's *User's Manual*).
- Automatic Update facility (see "Automatic Update Facility on page 39" on page 21).

Notes:

- BootP/TFTP is not applicable to MSBG Series.
- When configuring the device using the Web interface, loading an *ini* file is unnecessary. There is also no need for a TFTP server.
- When configuring the device using SNMP, the only configuration in the *ini* file is the IP address for the SNMP traps.
- For information on the structure of the *ini* file, refer to the device's *User's Manual*.

2.4 Using BootP / DHCP

Note: This section is not applicable to MSBG Series, Mediant 800 Gateway & E-SBC, Mediant 1000 Gateway & E-SBC, Mediant 3000 HA, and Mediant 4000.

The device uses the Bootstrap Protocol (BootP) and the Dynamic Host Configuration Protocol (DHCP) to automatically obtain its networking parameters and configuration after it is reset. BootP and DHCP are also used to provide the IP address of a TFTP server on the network and files (*cmp* and *ini*) to be loaded into memory.

Both DHCP and BootP are network protocols that enable a device to discover its assigned IP address. DHCP differs from BootP in that it provides a time-limited "lease" to the assigned address. Both protocols have been extended to enable the configuration of additional parameters specific to the device.

Note: BootP is typically used to initially configure the device. Thereafter, BootP is no longer required as all parameters can be stored in the device's non-volatile memory and used when BootP is inaccessible. BootP can be used later to change the device's IP address. For a description on using the BootP application, see "BootP/TFTP Configuration Utility" on page 183.

2.4.1 BootP/DHCP Server Parameters

BootP and DHCP can be used to provision the following parameters (included in the BootP/DHCP reply):

- **IP Address, Subnet Mask**: Mandatory parameters sent to the device every time a BootP/DHCP process occurs.
- **Default Gateway IP Address:** Optional parameter sent to the device only if configured in the BootP/DHCP server.
- **TFTP Server IP Address:** Optional parameter containing the IP address of the TFTP server from which the software (*cmp*) and *ini* files are loaded.
- **DNS Server IP Address (Primary and Secondary):** Optional parameters containing the IP addresses of the primary and secondary DNS servers. These parameters are available only for DHCP.
- **Syslog Server IP Address:** Optional parameter sent to the device only if configured. This parameter is available only for DHCP.
- **SIP Server IP Address:** Two optional parameters (primary and secondary SIP server) sent to the device only if configured. These parameters are available only for DHCP.
- **Firmware File Name:** Optional parameter containing the name of the firmware file to be loaded to the device using TFTP.
- **Configuration ini File Name:** Optional parameter containing the name of the *ini* file (proprietary configuration file with the extension *.ini) to be loaded to the device using TFTP. When the device detects that this parameter field is defined in BootP, it initiates a TFTP process to load the file to the device. The new configuration contained in the *ini* file can be stored in the device's integral non-volatile memory. Whenever the device is reset and no BootP reply is received, or the *ini* file name is missing in the BootP reply, the device uses the previously stored *ini* file.

2.4.2 DHCP Host Name Support

When the device is configured to use DHCP (using the *ini* file parameter DHCPEnable = 1), it attempts to contact the local DHCP server to obtain the networking parameters (IP address, subnet mask, default gateway, primary/secondary DNS server, and two SIP server addresses). These network parameters have a 'time limit'. After the time limit expires, the device must 'renew' its lease from the DHCP server.

To detect the device's IP address, follow one of the procedures below:

- 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 equal to the last six digits of the MAC address converted to 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 a URL of http://acl_<serial number> (instead of using the device's IP address). For example, if the device's MAC address is 00908f010280, the DNS name is acl_66176.
- After physically resetting the device, its IP address is displayed in the 'Client Info' column in the BootP/TFTP configuration utility (see "BootP/TFTP Configuration Utility" on page 183).
- Use serial communication software.
- Contact your System Administrator.

Notes:

- 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 happens while calls are in progress, they are not automatically rerouted to the new network address (since this function is beyond the scope of a VoIP device). Therefore, administrators are advised to configure DHCP servers to allow renewal of IP addresses.
- If, during operation, the device's IP address is changed as a result of a DHCP renewal, the device is automatically reset.
- If the device's network cable is disconnected and reconnected, a DHCP renewal is performed (to verify that the device is still connected to the same network). When DHCP is enabled, 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, DHCP 'option 60' is contained. If the device is reset from the Web interface or SNMP, only a single DHCP sequence containing 'option 60' is sent. If DHCP procedure is used, the new device IP address, allocated by the DHCP server, must be detected.

2.4.3 Microsoft DHCP/BootP Server

The device can be configured using any third-party BootP server, including Microsoft™ Windows™ DHCP server, to provide the device with an IP address and other initial parameter configurations.

To configure the Windows DHCP Server for assigning an IP address to BootP clients (i.e., device>s), add a Reservation for each BootP client. The Reservation builds an association between the MAC address (12 digits) provided in the accompanying device documentation and the IP address. Windows Server provides the IP address based on the device's MAC address in the BootP request frame. For information on how to add a reservation, view the "Managing Client Reservations Help" topic in the DHCP console.

To configure Windows DHCP server to provide Boot File information to BootP clients, edit the BootP Table in the DHCP console. The BootP Table must be enabled from the **Action** > **Properties** dialog box (select the option 'Show the BootP Table Folder' and then click **OK**). For information on editing the BootP Table, view the "Manage BOOTP and remote access clients" Help topic in the DHCP console.

The following parameters must be specified:

- Local IP address - the device’s IP address
- Subnet mask
- Gateway IP address - default Gateway IP address
- BootP File name - Optional (see the following Note)

Note: The BootP File field is typically not used. This field is only used for software upgrade.

2.4.4 Using BootP

Note: For a description on using the BootP application, see "BootP/TFTP Configuration Utility" on page 183.

2.4.4.1 Upgrading the Device

When upgrading the device (loading new software to the device) using the BootP/TFTP configuration utility:

- From version 4.4 to version 4.4 or to any higher version, the device retains its configuration (*ini* file). However, the auxiliary files (CPT, logo, etc.) may be erased.
- From version 4.6 to version 4.6 or to any higher version, the device retains its configuration (*ini* file) and auxiliary files (CPT, logo, etc.).

You can also use the Web interface's Software Upgrade Wizard to upgrade the device (refer to the device's *User's Manual*).

Note: To save the *cmp* file to the device's non-volatile memory, use the **-fb** command line switch. If the file is not saved, the device reverts to the old software version after the next reset. For information on using command line switches, see "Using Command Line Switches" on page 193.

2.4.4.2 Vendor Specific Information Field

The device uses the Vendor Specific Information field in the BootP request to provide device-related initial startup information. The BootP/TFTP utility displays this information in the Log window's 'Client Info' column (see "BootP/TFTP Configuration Utility" on page 183).

Note: This option is not available on DHCP servers.

The Vendor Specific Information field is disabled by default. To enable this feature, configure the *ini* file parameter ExtBootPReqEnable (refer to the device's *User's Manual*) or use the **-be** command line switch (see "BootP/TFTP Configuration Utility" on page 183).

The following table details the Vendor Specific Information field according to device:

Table 2-1: Vendor Specific Information Field

Tag #	Description	Value	Length
220	Device Type	- **#24** = TP-8410; IPM-8410 - **#02** = TP-1610; IPM-1610; Mediant 2000; IPmedia 2000 - **#08** = TP-6310; IPM-6310 - **#09** = IPmedia 3000; Mediant 3000; Mediant 1000; Mediant 600 - **#13** = MP-124 - **#14** = MP-118 - **#15** = MP-114 - **#16** = MP-112	1
221	Current IP Address	XXX.XXX.XXX.XXX	4
222	Burned Boot Software Version	X.XX	4
223	Burned *cmp* Software Version	XXXXXXXXXXXX	12
224	Geographical Address	0-31	1
225	Chassis Geographical Address	0-31	1
228	Indoor / Outdoor	- **#0** = Indoor - **#1** = Outdoor **Notes:** - Applicable only to analog interfaces. - Indoor is applicable only to FXS interfaces. - Outdoor is applicable only to FXO interfaces.	1
230	Analog Channels	- **#2** = MediaPack only - **#4**, **#8**, **#24** = Mediant 1000 and MediaPack - **#12**, **#16**, **#20** = Mediant 1000 only **Note:** Applicable only to analog interfaces.	1

The following table shows an example of the structure of the vendor specific information field:

Table 2-2: Structure of the Vendor Specific Information Field

Field	Value
Vendor-Specific Information Code	42
Length Total	12
Tag Num	220
Length	1
Value	2
Tab Num	225

Field	Value
Length	1
Value	1
Tag Num	221
Length	4
Value (1)	10
Value (2)	2
Value (3)	70
Value (4)	1
Tag End	255

2.4.4.3 Selective BootP

The Selective BootP mechanism allows the integral BootP client to filter out unsolicited BootP replies. This can be beneficial for environments where more than one BootP server exists and only one BootP server is used to configure AudioCodes devices. The command line switch **-bs** is used to activate this feature (see "Using Command Line Switches" on page 193).

2.5 Automatic Update Mechanism

The device can automatically update its *cmp*, *ini*, and auxiliary files. These files can be stored on any standard Web, FTP, or NFS server and can be loaded periodically to the device using HTTP, HTTPS, FTP, or NFS. This mechanism can be used even for devices that are installed behind NAT and firewalls. The Automatic Update mechanism is applied per file. For a detailed description on automatic configuration, see "Automatic Device Configuration" on page 135.

The Automatic Update mechanism is activated by the following:

- Upon device start-up (see "Startup Process" on page 13).
- At a user-defined time of day (e.g., 18:00), using the *ini* file parameter AutoUpdatePredefinedTime (disabled by default).
- At fixed intervals (e.g., every 60 minutes), using the *ini* file parameter AutoUpdateFrequency (disabled by default).
- Upon start-up, but before the device is operational, if the Secure Startup feature is enabled (see "Loading Files Securely (Disabling TFTP)" on page 139).

Notes:

- Mediant 800 MSBG and Mediant 1000 MSBG support only HTTP.
- If you implement the Automatic Update mechanism, the device must not be configured using the Web interface. If you configure parameters in the Web interface and save (burn) the new settings to the device's flash memory, the IniFileURL parameter (defining the URL to the ini file for Automatic Updates) is automatically set to 0 (i.e., Automatic Updates is disabled).
  The Web interface provides a safeguard for the Automatic Update mechanism. If the IniFileURL parameter is defined with a URL value (i.e., Automatic Updates is enabled), then by default, the 'Burn To FLASH' field under the Reset Configuration group in the Web interface's 'Maintenance Actions' page is automatically set to "No". Therefore, this prevents an unintended burn-to-flash when resetting the device. However, if configuration settings in the Web interface were burnt to flash, you can re-instate the Automatic Update mechanism, by loading to the device the ini file that includes the correct IniFileURL parameter setting, using the Web interface or BootP.
- The Automatic Update mechanism assumes that the external 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, adjust the update frequency (using the parameter AutoUpdateFrequency).
- When HTTP or HTTPS is used, the device queries the Web server(s) for the requested files. The *ini* file is loaded only if it was modified since the last automatic update. The *cmp* file is loaded only if its version is different from the version currently stored on the device's non-volatile memory. All other auxiliary files (e.g., CPT) are updated only once. To update a previously loaded auxiliary file, you must update the parameter containing its URL.
- To load different configurations (*ini* files) for specific devices, add the string '<MAC>' to the URL. This mnemonic is replaced with the device's hardware MAC address, resulting in an *ini* file name request that contains the device's MAC address.
- To automatically update the *cmp* file, use the parameter CmpFileURL to specify its name and location. As a precaution (to protect the device from accidental update), by default, the Automatic Update mechanism doesn't apply to the *cmp* file. Therefore, to enable it, set the parameter AutoUpdateCmpFile to 1.
- By default, when using the Automatic Update mechanism to load an *ini* file, a device reset is not performed automatically. If the *ini* file contains tables or parameters which are not applied on-the-fly (i.e., a device reset is required), the *ini* file must include ResetNow = 1 to initiate a device reset.
- By default, when using the Automatic Update mechanism to load an *ini* file, all parameters that are not included in the file are set to their default values. However, it is possible to configure only the parameters included in the *ini* file while retaining the settings of the other parameters (not included in the ini file). To achieve this, the *ini* file must include the parameter SetDefaultOnINIFileProcess = 0.

The following *ini* file example can be used to activate the Automatic Update mechanism.

```
# DNS is required for specifying domain names in URLs
[ InterfaceTable ]
FORMAT InterfaceTable_Index = InterfaceTable_ApplicationTypes,
InterfaceTable_InterfaceMode, InterfaceTable_IPAddress,
InterfaceTable_PrefixLength, InterfaceTable_Gateway,
InterfaceTable_VlanID, InterfaceTable_InterfaceName,
InterfaceTable_PrimaryDNSServerIPAddress,
InterfaceTable_SecondaryDNSServerIPAddress,
InterfaceTable_UnderlyingInterface;
InterfaceTable 0 = 6, 10, 10.13.4.12, 16, 10.13.0.1, 1, Mng,
10.1.1.11, 0.0.0.0, ;
[ \InterfaceTable ]
# Load an extra configuration ini file using HTTP
IniFileURL = 'http://webserver.corp.com/Gateway/inifile.ini'
# Load Call Progress Tones file using HTTPS
CptFileUrl = 'https://10.31.2.17/usa_tones.dat'
# Load Voice Prompts file using FTPS with user ‘root’ and password
‘wheel’
VPFileUrl = 'ftps://root:wheel@ftpserver.corp.com/vp.dat'
# Update every day at 03:00 AM
AutoUpdatePredefinedTime = '03:00'
# Note: The cmp file isn’t updated since it's disabled by default
(AutoUpdateCmpFile).
```

The following example illustrates how to utilize Automatic Updates for deploying a device with minimum manual configuration.

➢ To utilize Automatic Updates for deploying the device with minimal manual configuration:

1. Setup a Web server (e.g., http://www.corp.com) where all configuration files are located.

2. For each device, pre-configure the following parameter (DHCP / DNS are assumed):

```
IniFileURL = 'http://www.corp.com/master_configuration.ini'
```

3. Create a file named *master_configuration.ini* with the following text:

```
# Common configuration for all devices
# ------------------------------------
CptFileURL = 'http://www.corp.com/call_progress.dat'

# Check for updates every 60 minutes
AutoUpdateFrequency = 60

# Additional configuration per device
# -----------------------------------
# Each device loads a file named after its MAC address,
# (e.g., config_00908F033512.ini)
IniFileURL = 'http://www.corp.com/config_<MAC>.ini'

# Reset the device after configuration is updated.
# The device resets after all of the files are processed.
ResetNow = 1
```

You can modify the master_configuration.ini file (or any of the config_<MAC>.ini files) at any time. The device queries for the latest version every 60 minutes and applies the new settings immediately.

4. 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> for the Automatic Update mechanism.

5. The configuration URL can be provided using the Voice Configuration Menu (refer to the *User's Manual*). (Applicable only to Analog devices.)

6. To load configuration files from an NFS server, the NFS file system parameters should be defined in the configuration *ini* file. The following is an example of an *ini* file for loading files from NFS servers using NFS version 2.

```
# Define NFS servers for Automatic Update

[ NFSServers ]
FORMAT NFSServers_Index = NFSServers_HostOrIP,
NFSServers_RootPath, NFSServers_NfsVersion;
NFSServers 1 = 10.31.2.10, /usr/share, 2 ;
NFSServers 2 = 192.168.100.7, /d/shared, 2 ;
[ \NFSServers ]

CptFileUrl = 'file://10.31.2.10/usr/share/public/usa_tones.dat'

VpFileUrl =
'file://192.168.100.7/d/shared/gateways/voiceprompt.dat'
```

3 Command-Line Interface Based Management

Notes:

- This section is applicable only to non-MSBG devices.
- Full configuration through CLI is supported only by Mediant 800 MSBG and Mediant 1000 MSBG. For more information, refer to the following documents: *MSBG Series System & VoIP CLI Reference Guide* and *MSBG Series Data CLI Reference Guide*. For all other devices, CLI is used only for debugging and monitoring.

The command line interface (CLI) is available through a Telnet or a Secure SHell (SSH) session with the device's management interface. It mainly allows you to view various information regarding device setup and performance.

3.1 Starting a CLI Management Session

The procedure below describes how to start a CLI session.

➢ To start a CLI management session:

1. Enable CLI (Telnet or SSH), using one of the following methods:
   - **Web interface:**
     a. Open the ‘Telnet/SSH Settings' page (**Configuration** tab > **System** menu > **Management** submenu > **Telnet/SSH Settings**).
     b. From the ‘Embedded Telnet Server’ drop-down list, select ‘Enable (Unsecured)’ or ‘Enable Secured (SSL)’.
     c. From the ‘SSH Server Enable’ drop-down list, select ‘Enable’.
   - **ini file:** Configure the following *ini* file parameters as shown below:

     ```
     TelnetServerEnable = 1
     SSHServerEnable = 1
     ```

   - **SNMP:** set the objects acSysTelnetSSHServerEnable and acSysTelnetServerEnable to 'enable' (1).
2. Establish a Telnet or SSH session with the device's OAMP IP address.
3. Login to the session using the same username and password assigned to the Admin user of the Web interface.

   A Telnet or SSH client application must be running on the management PC. Most operating systems, including Microsoft Windows, include a built-in Telnet client, which can be activated from the command prompt. SSH, however, must be installed separately. See the following link for a discussion of available SSH client implementations: http://en.wikipedia.org/wiki/Comparison_of_SSH_clients.

After logging in, the current directory (root), available commands (**SHow**, **PING**), available subdirectories, and a welcome message are displayed at the CLI prompt:

```
login: Admin
password:
GW device ready. Type "exit" to close the connection.
MGmt/ CONFiguration/ IPNetworking/ TPApp/ BSP/
SHow PING
/>
```

Notes:

- By default, CLI access is disabled for security.
- The user name and password are case-sensitive.
- Only the primary User Account (which has Security Administration access level - 200) can be used to access the device using Telnet/CLI. This user is defined in the device's Web interface.
- The CLI user name and password can be changed by the device's administrator. Multiple users can be defined.

3.2 CLI Navigation Concepts

Commands are organized in subdirectories. When the CLI session starts, you are located in the 'root' directory, which contains only two commands: **SHow** and **PING**. To access a subdirectory, type its name, and then press <Enter>. To move back one directory, type two periods (..), and then press <Enter>. Alternatively, if you know the full path to a command inside one of the subdirectories, the short format can be used to run it directly. For example, the **PERFormance** command in the MGmt subdirectory may be run directly by typing **/mg/perf**.

The CLI commands can be entered in an abbreviated format by typing only the letters shown in upper case (i.e., capital letters). For example, the **CHangePassWord** command can be entered by typing **chpw**.

3.3 Commands

The following table summarizes the CLI commands and their options.

Table 3-1: Summary of CLI Commands

Purpose	Commands	Description
Help	**h**	Displays the help for a specific command, action, or parameter.
Navigation	**cd**	Enters another directory.
	cd root	Navigates to the root directory (/).
	..	Goes up one level.
	exit	Terminates the CLI session.
Status	**show**	Displays the device's operational status.

Purpose	Commands	Description
	ping	Sends Internet Control Message Protocol (ICMP) echo request packets from the device to a defined IP address.
Configuration	**/conf/scp**	Sets a value for the specific parameter.
	/conf/rfs	Restores factory defaults.
	/conf/sar	Restarts the device.

3.3.1 General Commands

The following table summarizes the General commands and their corresponding options.

Table 3-2: General CLI Commands

Command	Short Format	Arguments	Description
SHow	sh	info | tdm | dsp | ip | log	Displays operational data. The arguments are documented below.
SHow INFO	sh info	-	Displays device hardware information, versions, uptime, temperature reading, and the last reset reason.
SHow HW	sh hw	--	Displays Mediant 3000 system information: power status, High-Availability status, and fan information.
SHow TDM	sh tdm	status | perf | summary	Displays the alarm status and performance statistics for E1/T1 trunks.
SHow DSP	sh dsp	status | perf	Displays status and version for each DSP device, along with overall performance statistics.
SHow IP	sh ip	conf | perf | route	Displays IP interface status and configuration, along with performance statistics. **Note:** The display format may change according to the configuration.
SHow LOG	sh log	[stop]	Displays (or stops displaying) Syslog messages in the CLI session.
/SIP/TestCall	/sip/tc	set dest | set src | set cid | set DTMFs | display | connect	Simulates an IP-to-PSTN call. See "Test Call (TC) Commands" on page 31 for additional details. **Note:** Only one test call can be activated at a given time.
PING	ping	[-n count] [-l size] [-w timeout] [-p cos] ip-address	Sends ICMP echo request packets to a specified IP address. ▪ count: number of packets to send. ▪ size: payload size in each packet. ▪ timeout: time (in seconds) to wait for a reply to each packet.

Command	Short Format	Arguments	Description
			- cos: Class-of-Service (as per 802.1p) to use.

Example:

```
/>sh ?
Usage:
   SHow INFO           Displays general device information
   SHow TDM            Displays PSTN-related information
   SHow DSP            Displays DSP resource information
   SHow IP             Displays information about IP interfaces

/>sh info
Board type: gateway SDH, firmware version 5.80.000.020
Uptime: 0 days, 0 hours, 3 minutes, 54 seconds
Memory usage: 63%
Temperature reading: 39 C
Last reset reason:
Board was restarted due to issuing of a reset from Web interface
Reset Time : 7.1.2009 21.51.13
/>sh tdm status
Trunk 00:  Active
Trunk 01:  Active
Trunk 02:  Active
Trunk 03:  Active
Trunk 04:  Active
Trunk 05:  Active
Trunk 06:  Active
Trunk 07:  Active
Trunk 08:  Active
Trunk 09:  Active
Trunk 10:  Active
Trunk 11:  Active
Trunk 12:  Active
Trunk 13:  Active
Trunk 14:  Active
Trunk 15:  Not Configured
Trunk 16:  Not Configured
Trunk 17:  Not Configured
Trunk 18:  Not Configured
Trunk 19:  Not Configured
Trunk 20:  Not Configured
Trunk 21:  Not Configured
/>sh tdm perf
DS1 Trunk Statistics (statistics for 948 seconds):
Trunk #    B-Channel  Call count RTP packet RTP packet Activity
           utilization           Tx         Rx        Seconds
0          1          1          2865       0          57
1          0          0          0          0          0
2          20         20         149743     0          3017
3          0          0          0          0          0
4          0          0          0          0          0
5          0          0          0          0          0
6          0          0          0          0          0
7          0          0          0          0          0
8          0          0          0          0          0
```

```
9          0          0          0          0          0
10         0          0          0          0          0
11         0          0          0          0          0
12         0          0          0          0          0
13         0          0          0          0          0
14         0          0          0          0          0
/>sh dsp status
DSP firmware: 491096AE8 Version:0540.03 Used=0 Free=480 Total=480
DSP device  0:  Active    Used=16   Free= 0   Total=16
DSP device  1:  Active    Used=16   Free= 0   Total=16
DSP device  2:  Active    Used=16   Free= 0   Total=16
DSP device  3:  Active    Used=16   Free= 0   Total=16
DSP device  4:  Active    Used=16   Free= 0   Total=16
DSP device  5:  Active    Used=16   Free= 0   Total=16
DSP device  6:  Inactive
DSP device  7:  Inactive
DSP device  8:  Inactive
DSP device  9:  Inactive
DSP device 10:  Inactive
DSP device 11:  Inactive
DSP device 12:  Active    Used=16   Free= 0   Total=16
DSP device 13:  Active    Used=16   Free= 0   Total=16
DSP device 14:  Active    Used=16   Free= 0   Total=16
DSP device 15:  Active    Used=16   Free= 0   Total=16
DSP device 16:  Active    Used=16   Free= 0   Total=16
DSP device 17:  Active    Used=16   Free= 0   Total=16
DSP device 18:  Inactive
PSEC - DSP firmware: AC491IPSEC Version: 0540.03
CONFERENCE - DSP firmware: AC491256C Version: 0540.03
/>sh dsp perf
DSP Statistics (statistics for 968 seconds):
Active DSP resources: 480
Total DSP resources: 480
DSP usage %: 100
/>sh ip perf
Networking Statistics (statistics for 979 seconds):
IP KBytes TX: 25
IP KBytes RX: 330
IP KBytes TX per second: 0
IP KBytes RX per second: 1
IP Packets TX: 1171
IP Packets RX: 5273
IP Packets TX per second: 3
IP Packets RX per second: 12
Peak KByte/s TX in this interval: 18
Peak KByte/s RX in this interval: 4
Discarded packets: 186
DHCP requests sent: 0
IPSec Security Associations: 0
/>/mg/perf reset
Done.
/>sh ip perf
Networking Statistics (statistics for 2 seconds):
IP KBytes TX: 2
IP KBytes RX: 4
IP KBytes TX per second: 0
IP KBytes RX per second: 1
IP Packets TX: 24
IP Packets RX: 71
IP Packets TX per second: 3
IP Packets RX per second: 12
Peak KByte/s TX in this interval: 18
Peak KByte/s RX in this interval: 4
Discarded packets: 0
DHCP requests sent: 0
IPSec Security Associations: 0

/>sh ip conf
Interface  IP Address          Subnet Mask     Default Gateway
---------  ------------------   --------------   ----------
 OAM           10.4.64.13     55.255.0.0           10.4.0.1
 Media         10.4.64.13     255.255.0.0          10.4.0.1
 Control       10.4.64.13     255.255.0.0          10.4.0.1
MAC address: 00-90-8f-04-5c-e9

/>sh ip route
Destination      Mask             Gateway          Intf Flags

------------- ---------------- ------------------------
0.0.0.0          0.0.0.0          10.4.0.1         OAM  A  S
10.4.0.0         255.255.0.0      10.4.64.13       OAM  A L
127.0.0.0        255.0.0.0        127.0.0.1        AR S
127.0.0.1        255.255.255.255  127.0.0.1        A L    H
Flag legend: A=Active R=Reject L=Local S=Static E=rEdirect
M=Multicast
             B=Broadcast H=Host I=Invalid
End of routing table, 4 entries displayed.

/>ping 10.31.2.10
Ping process started for address 10.31.2.10. Process ID - 27.
Reply from 10.31.2.10: bytes=0 time<0ms
Reply from 10.31.2.10: bytes=0 time<0ms
Reply from 10.31.2.10: bytes=0 time<0ms
Reply from 10.31.2.10: bytes=0 time<0ms

Ping statistics for 10.31.2.10:
Packets:Sent = 4, Received = 4, Lost 0 (0% loss),
Approximate round trip times in milli-seconds:
Minimum = 0ms, Maximum = 0ms, Average = 0ms
```

3.3.2 Configuration Commands

The commands under the 'CONFiguration' directory, query and modify the current device configuration. The following commands are available:

Table 3-3: Configuration CLI Commands

Command	Short Format	Arguments	Description
SetConfigParam IP	/conf/scp ip	ip-addr subnet def-gw	Sets the IP address, subnet mask, and default gateway address of the device (on-the-fly). **Note:** This command may cause disruption of service. The CLI session may disconnect since the device changes its IP address.

Command	Short Format	Arguments	Description
RestoreFactorySettings	/conf/rfs		Restores all parameters to factory settings.
SaveAndRestart	/conf/sar		Saves all current configurations to the non-volatile memory and resets the device.
ConfigFile	/conf/cf	view | get | set	Retrieves the full *ini* file from the device and allows loading a new *ini* file directly in the CLI session. **Note:** The argument *view* displays the file, page by page. The argument *get* displays the file without breaks.

3.3.3 Test Call (TC) Commands

The command **/sip/tc** can be used to simulate an IP-to-PSTN call. The following sub-commands are available:

Table 3-4: Sub-Commands of Test (TC) Command

Sub-Command	Arguments	Description
set dest	<number>	Sets the Destination Number for the test call.
set src	<number>	Sets the Source Number for the test call.
set cid	<display string>	Sets the Display Name for the test call.
set DTMFs	<DTMF pattern>	Sets the pattern of DTMFs that is played to the PSTN side after the test call is connected.
display		Displays all the set parameters for the test call. These values can be manually set using the **set** commands or defaults.
connect	<destination number> <source number> <caller ID> <time>	Generates the test call toward the PSTN side using the set parameters. If no arguments are defined in the **connect** command line, the test call uses the values defined using the **set** commands (or defaults). **Note:** There is no option for defining only the <time> value without specifying all the other arguments that appear before it.

Example:

```
/GWApp/TestCall>tc ?
TestCall - TC
Manage Test Tel-Link Call.
Usage:
      TC set dest <number>
      TC set src  <number>
      TC set cid  <display string>
      TC set DTMFs  <dtmf string>
      TC display
      TC connect [dest-number] [src number] [CID] [time]
```

```
TestCall
/GWApp/TestCall>tc display
Test Call configuration
      Dest number: 402
      Src  number: 700
      Cid display: TESTING
      DTMFs String 112233
      Time After DTMF are sent: 20 sec
TestCall
/GWApp/TestCall>tc connect
Start Test call.
Receive ALERT Event
Receive Connect
Wait more 20 seconds before disconnecting
Receive SS_TIMER_EV Event
Send Release To Call
Receive RELEASE_ACK Event
Receive SS_TIMER_EV Event
TestCall
/GWApp/TestCall>
```

3.3.4 Management Commands

The commands under the 'MGmt' directory, described in the table below, display current performance values.

Table 3-5: CLI Management Command

Command	Short Format	Arguments	Description
/MGmt/PERFormance	/mg/perf	basic | control | dsp | net | ds1 | reset	Displays performance statistics. The *reset* argument clears all statistics to zero.

3.3.5 PSTN Commands

The commands under the 'PSTN' directory allow you to perform various PSTN actions.

Note: PSTN CLI commands are applicable only to Digital PSTN devices.

Table 3-6: PSTN CLI Command

Command	Short Format	Arguments	Description
PstnLoopCommands	PS/PH/PLC	<TrunkId> <LoopCode> <BChannel>	Activates a loopback on a specific trunk and B-channel. For loopback on the entire trunk, set BChannel=(-1). The valid value options for LoopCode include the following:

Command	Short Format	Arguments	Description
			▪ 0 = NO_LOOPS ▪ 1 = REMOTE_LOOP (whole trunk only - not applicable to BRI) ▪ 2 = LINE_PAYLOAD_LOOP (whole trunk only - not applicable to BRI, DS3 or Sonet/SDH) ▪ 3 = LOCAL_ALL_CHANNELS_LOOP (whole trunk only - not applicable to DS3 or Sonet/SDH)
PstnSendAlarm	PS/PH/PSA	<TrunkId> <AlarmSendCode>	Sends an alarm signal at the Tx interface or on a specific Trunk ID. The valid value options for AlarmSendCode include the following: ▪ 0 = NO_ALARMS (means stop sending AIS) ▪ 1 = AIS_ALARM ▪ 2 = STOP_RAI_ALARM ▪ 3 = SEND_RAI_ALARM
DeleteCasFile	PS/CAS/DCF	<TableIndex>	Deletes all the device's CAS files when <TableIndex> is set to -1 (other options are currently not supported).

3.3.6 LDAP Commands

The commands under the 'IPNetworking\OpenLdap' directory allow you to perform various Lightweight Directory Access Protocol (LDAP) actions.

Table 3-7: LDAP Commands

Sub-Command	Arguments	Description
LdapSTatus	-	Displays the LDAP connection status.
LdapSearch	<search key> <attribute1> [<attribute 2> ...<attribute 5>]	Searches an LDAP server. The parameters enclosed by [] are optional.
LDapOpen	-	Opens a connection to the LDAP server using parameters provided in the configuration file.
LDapSetDebugmode	<mode>	Sets the LdapDebugLevelMode parameter. Possible levels 0-3.
LDapGetDebugmode	-	Gets the LdapDebugLevelMode parameter value

4 SNMP-Based Management

Simple Network Management Protocol (SNMP) is a standards-based network control protocol for managing elements in a network. The SNMP Manager (usually implemented by a network Management System (NMS) or an Element Management System (EMS) connects to an SNMP Agent (embedded on a remote Network Element (NE)) to perform network element Operation, Administration, Maintenance, and Provisioning (OAMP).

Both the SNMP Manager and the NE refer to the same database to retrieve information or configure parameters. This database is referred to as the Management Information Base (MIB), and is a set of statistical and control values. Apart from the standard MIBs documented in IETF RFCs, SNMP additionally enables the use of proprietary MIBs, containing non-standard information set (specific functionality provided by the Network Element).

Directives, issued by the SNMP Manager to an SNMP Agent, consist of the identifiers of SNMP variables (referred to as MIB object identifiers or MIB variables) along with instructions to either get the value for that identifier, or set the identifier to a new value (configuration). The SNMP Agent can also send unsolicited events towards the EMS, called SNMP traps.

The definitions of MIB variables supported by a particular agent are incorporated in descriptor files, written in Abstract Syntax Notation (ASN.1) format, made available to EMS client programs so that they can become aware of MIB variables and their usage.

The device contains an embedded SNMP Agent supporting both general network MIBs (such as the IP MIB), VoP-specific MIBs (such as RTP) and proprietary MIBs (acGateway, acAlarm, acMedia, acControl, and acAnalog MIBs) enabling a deeper probe into the interworking of the device. All supported MIB files are supplied to customers as part of the release.

4.1 SNMP Standards and Objects

This section discusses the SNMP standards and SNMP objects.

4.1.1 SNMP Message Standard

Four types of SNMP messages are defined:

- **Get:** A request that returns the value of a named object.
- **Get-Next:** A request that returns the next name (and value) of the "next" object supported by a network device given a valid SNMP name.
- **Set:** A request that sets a named object to a specific value.
- **Trap:** A message generated asynchronously by network devices. It notifies the network manager of a problem apart from the polling of the device.

Each of these message types fulfills a particular requirement of network managers:

- **Get Request:** Specific values can be fetched via the "get" request to determine the performance and state of the device. Typically, many different values and parameters can be determined via SNMP without the overhead associated with logging into the device, or establishing a TCP connection with the device.
- **Get Next Request:** Enables the SNMP standard network managers to "walk" through all SNMP values of a device (via the "get-next" request) to determine all names and values that a device supports.

- **Get-Bulk:** Extends the functionality of GETNEXT by allowing multiple values to be returned for selected items in the request.
- This is accomplished by beginning with the first SNMP object to be fetched, fetching the next name with a "get-next", and repeating this operation.
- **Set Request:** The SNMP standard provides a action method for a device (via the "set" request) to accomplish activities such as disabling interfaces, disconnecting users, clearing registers, etc. This provides a way of configuring and controlling network devices via SNMP.
- **Trap Message:** The SNMP standard furnishes a mechanism for a device to "reach out" to a network manager on their own (via the “trap" message) to notify or alert the manager of a problem with the device. This typically requires each device on the network to be configured to issue SNMP traps to one or more network devices that are awaiting these traps.

The above message types are all encoded into messages referred to as "Protocol Data Units" (PDUs) that are interchanged between SNMP devices.

4.1.2 SNMP MIB Objects

The SNMP MIB is arranged in a tree-structure, similar to a disk directory structure of files. The top level SNMP branch begins with the ISO "internet" directory, which contains four main branches:

- **"mgmt" SNMP branch:** Contains the standard SNMP objects usually supported (at least in part) by all network devices.
- **“private" SNMP branch:** Contains those "extended" SNMP objects defined by network equipment vendors.
- **"experimental" and "directory" SNMP branches:** Also defined within the "internet" root directory, are usually devoid of any meaningful data or objects.

The "tree" structure described above is an integral part of the SNMP standard, though the most pertinent parts of the tree are the "leaf" objects of the tree that provide actual management data regarding the device. Generally, SNMP leaf objects can be partitioned into two similar but slightly different types that reflect the organization of the tree structure:

- **Discrete MIB Objects:** Contain one precise piece of management data. These objects are often distinguished from "Table" items (below) by adding a “.0" (dot-zero) extension to their names. The operator must merely know the name of the object and no other information.
- **Table MIB Objects:** Contain multiple pieces of management data. These objects are distinguished from "Discrete" items (above) by requiring a "." (dot) extension to their names that uniquely distinguishes the particular value being referenced. The "." (dot) extension is the "instance" number of an SNMP object. For "Discrete" objects, this instance number is zero. For "Table" objects, this instance number is the index into the SNMP table. SNMP tables are special types of SNMP objects, which allow parallel arrays of information to be supported. Tables are distinguished from scalar objects, such that tables can grow without bounds. For example, SNMP defines the "ifDescr" object (as a standard SNMP object) that indicates the text description of each interface supported by a particular device. Since network devices can be configured with more than one interface, this object can only be represented as an array.

By convention, SNMP objects are always grouped in an "Entry" directory, within an object with a "Table" suffix. (The "ifDescr" object described above resides in the "ifEntry" directory contained in the "ifTable" directory).

4.1.3 SNMP Extensibility Feature

One of the principal components of an SNMP manager is a MIB Compiler, which allows new MIB objects to be added to the management system. When a MIB is compiled into an SNMP manager, the manager is made "aware" of new objects that are supported by agents on the network. The concept is similar to adding a new schema to a database.

Typically, when a MIB is compiled into the system, the manager creates new folders or directories that correspond to the objects. These folders or directories can typically be viewed with a "MIB Browser", which is a traditional SNMP management tool incorporated into virtually all network management systems.

The act of compiling the MIB allows the manager to know about the special objects supported by the agent and access these objects as part of the standard object set.

4.2 Carrier-Grade Alarm System

The basic alarm system has been extended to a carrier-grade alarm system. A carrier-grade alarm system provides a reliable alarm reporting mechanism that takes into account element management system (EMS) outages, network outages, and transport mechanism such as SNMP over UDP.

A carrier-grade alarm system is characterized by the following:

- The device allows an EMS to determine which alarms are currently active in the device. That is, the device maintains an active alarm table.
- The device allows an EMS to detect lost alarms and clear notifications. [sequence number in trap, current sequence number MIB object]
- The device allows an EMS to recover lost alarm raise and clear notifications [maintains a log history]
- The device sends a cold start trap to indicate that it is starting. This allows the EMS to synchronize its view of the device's active alarms.

When the SNMP alarm traps are sent, the carrier-grade alarm system does not add or delete alarm traps as part of the feature. This system provides the mechanism for viewing of history and current active alarm information.

4.2.1 Active Alarm Table

The device maintains an active alarm table to allow an EMS to determine which alarms are currently active in the device. Two views of the active alarm table are supported by the agent:

- acActiveAlarmTable in the enterprise AcAlarm
- alarmActiveTable and alarmActiveVariableTable in the IETF standard AcAlarm MIB (rooted in the MIB tree)

The acActiveAlarmTable is a simple, one-row per alarm table that is easy to view with a MIB browser.

4.2.2 Alarm History

The device maintains a history of alarms that have been raised and traps that have been cleared to allow an EMS to recover any lost raise or clear traps. Two views of the alarm history table are supported by the agent:

- acAlarmHistoryTable in the enterprise AcAlarm - a simple, one-row per alarm table, that is easy to view with a MIB browser.
- nlmLogTable and nlmLogVariableTable in the standard NOTIFICATION-LOG-MIB

4.2.3 SONET Alarm Consolidation

You can enable the device to send trunk alarms only on the DS3 level (instead of trunk level). When the DS3AlarmConsolidation parameter is set to 1, the PSTN alarms are consolidated. In such a setup, only SDH alarms are raised and no alarms are raised for trunks (even if they exist). When the SDH alarm is cleared, trunk alarms are raised (if they exist.

Note: This section is applicable only to Mediant 3000/TP-6310.

4.3 Topology MIB - Objects

4.3.1 Physical Entity - RFC 2737

The following groups are supported:

- **entityPhysical group:** Describes the physical entities managed by a single agent.
- **entityMapping group:** Describes the associations between the physical entities, logical entities, interfaces, and non-interface ports managed by a single agent.
- **entityGeneral group:** Describes general system attributes shared by potentially all types of entities managed by a single agent.
- **entityNotifications group:** Contains status indication notifications.

4.3.2 IF-MIB - RFC 2863

The following interface types are presented in the ifTable:

- **ethernetCsmacd(6):** for all Ethernet-like interfaces, regardless of speed, as per RFC 3635 (Gigabit Ethernet for 3000 Series devices)
- **ds1(18):** DS1-MIB
- **voiceFXO(101):** Voice Foreign Exchange Office. (Applicable only to MP-118 and Mediant 1000.)
- **voiceFXS(102):** Voice Foreign Exchange Station. (Applicable only to MP-118 and Mediant 1000.)
- **sonet(39):** SONET-MIB. (Applicable only to the 3000 Series.)

- **ds3(30):** DS3-MIB. (Applicable only to the 3000 Series.)

The numbers in the brackets above refer to the IANA's interface-number.

For each interface type, the following objects are supported:

Table 4-1: DS1 Digital Interfaces (Digital PSTN)

ifTable	Value
ifDescr	Digital DS1 interface.
ifType	ds1(18).
ifMtu	Constant zero.
ifSpeed	DS1 = 1544000, or E1 = 2048000, according to dsx1LineType
ifPhysAddress	The value of the Circuit Identifier [dsx1CircuitIdentifier]. If no Circuit Identifier has been assigned this object should have an octet string with zero length.
ifAdminStatus	Trunk's Lock & Unlock during run time. In initialization process we need to refer the Admin-Status parameter.
ifOperStatus	Up or Down, according to the operation status.
ifLastChange	The value of sysUpTime at the time the interface entered its current operational state.
ifXTable	**Value**
ifName	Digital# acTrunkIndex
ifLinkUpDownTrapEnable	Set to enabled(1)
ifHighSpeed	Speed of line in Megabits per second: 2
ifConnectorPresent	Set to true(1) normally, except for cases such as DS1/E1 over AAL1/ATM where false(2) is appropriate
ifCounterDiscontinuityTime	Always zero.

Table 4-2: BRI Interfaces (Applicable to MSBG Series, Mediant 1000 & Mediant 600)

ifTable	Value
ifDescr	BRI interface
ifType	isdns(75)
ifMtu	Constant zero
ifSpeed	144000
ifPhysAddress	Octet string with zero length
ifAdminStatus	Trunk's Lock & Unlock during run time. In initialization process, refer to the Admin-Status parameter.
ifOperStatus	Up or Down according to the operation status.
ifLastChange	The value of sysUpTime at the time the interface entered its current operational state.

ifTable	Value
ifDescr	BRI interface
ifXTable	**Value**
ifName	BRI port no. #
ifLinkUpDownTrapEnable	Set to enabled (1)
ifHighSpeed	Speed of line in megabits per second.
ifPromiscuousMode	Non promiscuous mode (1)
ifConnectorPresent	Set to true (1) normally
ifCounterDiscontinuityTime	Always zero

Table 4-3: Ethernet (Gigabit for 3000 Series) Interface

ifTable & ifXTable	Value
ifIndex	Constructed as defined in the device's Index format.
ifDescr	Ethernet interface.
ifType	ethernetCsmacd(6)
ifMtu	1500
ifSpeed	acSysEthernetFirstPortSpeed in bits per second (Applicable only to Mediant 1000) 0 since it's GBE - refer to ifHighSpeed (Applicable only to 3000 Series).
ifPhysAddress	00-90-8F plus acSysIdSerialNumber in hex.Will be same for both dual ports.
ifAdminStatus	Always UP. [Read Only] - Write access is not required by the standard. Support for 'testing' is not required.
ifOperStatus	Up or Down corresponding to acAnalogFxsFxoType where Unknown is equal to Down.
ifLastChange	The value of sysUpTime at the time the interface entered its current operational state.
ifInOctets	The number of octets in valid MAC frames received on this interface, including the MAC header and FCS. This does include the number of octets in valid MAC Control frames received on this interface.
ifInUcastPkts	As defined in IfMIB.
ifInDiscards	As defined in IfMIB.
ifInErrors	The sum for this interface of dot3StatsAlignmentErrors, dot3StatsFCSErrors, dot3StatsFrameTooLongs, and dot3StatsInternalMacReceiveErrors.
ifInUnknownProtos	As defined in IfMIB.
ifOutOctets	The number of octets transmitted in valid MAC frames on this interface, including the MAC header and FCS. This does include the number of octets in valid MAC

ifTable & ifXTable	Value
	Control frames transmitted on this interface.
ifOutUcastPkts	As defined in IfMIB.
ifOutDiscards	As defined in IfMIB.
ifOutErrors	The sum for this interface of: dot3StatsSQETestErrors, dot3StatsLateCollisions, dot3StatsExcessiveCollisions, dot3StatsInternalMacTransmitErrors and dot3StatsCarrierSenseErrors.
ifName	Ethernet (Gigabit for 3000 Series) port #1 or# 2
ifInMulticastPkts	As defined in IfMIB.
ifInBroadcastPkts	As defined in IfMIB.
ifOutMulticastPkts	As defined in IfMIB.
ifOutBroadcastPkts	As defined in IfMIB.
ifHCInOctets **ifHCOutOctets**	64-bit versions of counters. Required for ethernet-like interfaces that are capable of operating at 20 Mb/s or faster, even if the interface is currently operating at less than 20 Mb/s.
ifHCInUcastPkts **ifHCInMulticastPkts** **ifHCInBroadcastPkts** **ifHCOutUcastPkts** **ifHCOutMulticastPkts** **ifHCOutBroadcastPkts**	64-bit versions of packet counters. Required for ethernet-like interfaces that are capable of operating at 640 Mb/s or faster, even if the interface is currently operating at less than 640 Mb/s. Therefore, will be constant zero.
ifLinkUpDownTrapEnable	Refer to [RFC 2863]. Default is 'enabled'
ifHighSpeed	**3000 Series:** 1000 **Mediant 1000:** 10 or 100 according to acSysEthernetFirstPortSpeed
ifPromiscuousMode	Constant False. [R/O]
ifConnectorPresent	Constant True.
ifAlias	An 'alias' name for the interface as specified by a network manager (NVM)
ifCounterDiscontinuityTime	As defined in IfMIB.

Table 4-4: SONET /SDH Interfaces (Mediant 3000)

ifTable & ifXTable	Value
ifDescr	SONET/SDH interface. Module #n Port #n
ifType	sonet(39).
ifMtu	Constant zero.
ifSpeed	155520000
ifPhysAddress	The value of the Circuit Identifier. If no Circuit Identifier has been assigned this object should have an octet string with zero length.

ifTable & ifXTable	Value
ifAdminStatus	Read-only access -- Always UP.
ifOperStatus	The value testing(3) is not used. This object assumes the value down(2), if the objects sonetSectionCurrentStatus and sonetLineCurrentStatus have any other value than sonetSectionNoDefect(1) and sonetLineNoDefect(1), respectively.
ifLastChange	The value of sysUpTime at the time the interface entered its current operational state.
ifName	SONET /SDH port no. n
ifLinkUpDownTrapEnable	Set to enabled(1)
ifHighSpeed	Speed of line in Megabits per second: 155
ifConnectorPresent	Set to true(1) normally, except for cases such as DS1/E1 over AAL1/ATM where false(2) is appropriate
ifCounterDiscontinuityTime	Always zero.

Table 4-5: DS3 Interfaces (Mediant 3000)

ifTable & ifXTable	Value
ifDescr	DS3 interface, Module no.#d, Port no.#d
ifType	Ds3(30).
ifMtu	Constant zero.
ifSpeed	44736000
ifPhysAddress	The value of the Circuit Identifier. If no Circuit Identifier has been assigned this object should have an octet string with zero length.
ifAdminStatus	Read-only access -- Always UP.
ifOperStatus	The value testing(3) is not used. This object assumes the value down(2), if the objects dsx3LineStatus has any other value than dsx3NoAlarm(1).
ifLastChange	The value of sysUpTime at the time the interface entered its current operational state.
ifName	DS3 port no. n
ifLinkUpDownTrapEnable	Set to enabled(1)
ifHighSpeed	Speed of line in Megabits per second: 45
ifConnectorPresent	Set to true(1)
ifCounterDiscontinuityTime	Always zero.

4.4 Cold Start Trap

The device supports a cold start trap to indicate that the device is starting up. This allows the EMS to synchronize its view of the device's active alarms. In fact, two different traps are sent at start-up:

- **Standard coldStart trap:** iso(1).org(3).dod(6).internet(1). snmpV2(6). snmpModules(3). snmpMIB(1). snmpMIBObjects(1). snmpTraps(5). coldStart(1) sent at system initialization.
- **Enterprise acBoardEvBoardStarted:** generated at the end of system initialization. This is more of an "application-level" cold start sent after all the initializing process is over and all the modules are ready.

4.5 File Management

SNMP supports file download, upload, and removal.

4.5.1 Downloading a File to the Device

The file URL is set in the appropriate MIB object under the acSysHTTPClient subtree (refer to the subtree objects description for the URL form). The download can be scheduled using the acSysHTTPClientAutoUpdatePredefinedTime and acSysHTTPClientAutoUpdateFrequency objects. It can also be a manual process using acSysActionSetAutoUpdate. In this case (only) and as long as one URL is set at a time, the result can be viewed in acSysActionSetAutoUpdateActionResult. In both cases, the acHTTPDownloadResult trap is sent, indicating the success or failure of the process.

acSysActionSetActionId can be set to any value and can be used to indicate an action performed by a certain manager.

A successful process also ends with the file name in the appropriate object under the acSysFile subtree or in the acCASFileTable or the acAuxiliaryFiles subtree, along with the URL being erased from the object under the acSysHTTPClient subtree.

Notes:

- The action result (both in the acSysActionSetAutoUpdateActionResult object and acHTTPDownloadResult trap) for the Voice Prompt and XML indicates only that the file reached the device and has no indication on the application’s ability to parse the file.
- The action result in acSysActionSetAutoUpdateActionResult is reliable as long as only one file is downloaded at a time.

4.5.2 Uploading and Deleting a File

File upload is the procedure of sending a file from the device to the manager. Deleting a file is erasing it from the device, an offline action that requires a reset for it to be applied. The acSysUpload subtree holds all relevant objects.

acSysUploadFileURI indicates the file name and location along with the file transfer protocol (HTTP or NFS). For example, “http:\\server\filename.txt”.

acSysUploadFileType and acSysUploadFileNumber are used to determine the file to be uploaded along with its instance when relevant (for CAS or Video Font).

acSysUploadActionID is at the disposal of the manager and can be used to indicate that a certain manager has performed the action.

acSysUploadActionType determines the action that occurs and triggers it off at the same time.

Note: File upload using SNMP is supported only for ini files; file removal using SNMP is supported for all files except ini files.

4.6 Performance Measurements

Performance measurements are available for a third-party performance monitoring system through an SNMP interface. These can be polled at scheduled intervals by an external poller or utility in the management server or other off-board systems.

The device provides performance measurements in the form of two types:

- **Gauges:** Gauges represent the current state of activities on the device. Gauges unlike counters can decrease in value and like counters, can increase. The value of a gauge is the current value or a snapshot of the current activity on the device at that moment.
- **Counters:** Counters always increase in value and are cumulative. Counters, unlike gauges, never decrease in value unless the server is reset and then the counters are zeroed.

The device performance measurements are provided by several proprietary MIBs (located under the acPerformance subtree):

iso(1).org(3).dod(6).internet(1).private(4).enterprises(1).AudioCodes(5003).acPerformance(10).

The performance monitoring MIBs all have an identical structure, which includes two major subtrees:

- **Configuration**: allows configuration of general attributes of the MIB and specific attributes of the monitored objects
- **Data**

The monitoring results are presented in tables. There are one or two indices in each table. If there are two indices, the first is a sub-set in the table (e.g., trunk number) and the second (or a single where there is only one) index represents the interval number (present - 0, previous - 1, and the one before - 2).

The MIBs include:

- **acPMMedia**: media-related (voice) monitoring such as RTP and DSP.
- **acPMControl**: Control Protocol-related monitoring such as connections, commands.
- **acPMAnalog:** Analog channels off-hook state. (Applicable only to Analog devices.)
- **acPMPSTN:** PSTN-related monitoring such as channel use, trunk utilization. (Applicable only to Digital PSTN devices.)
- **acPMSystem:** general (system-related) monitoring.
- **acPMMediaServer:** for Media Server specific monitoring. (Applicable only to 3000 Series.)

The log trap acPerformanceMonitoringThresholdCrossing (non-alarm) is sent every time the threshold of a Performance Monitored object is crossed. The severity field is 'indeterminate' when the crossing is above the threshold and 'cleared' when it goes back under the threshold. The 'source' varbind in the trap indicates the object for which the threshold is being crossed.

4.6.1 Total Counters

The counter's attribute 'total' accumulates counter values since the device's most recent restart. The user can reset the total's value by setting the Reset-Total object.

Each MIB module has its own Reset Total object, as follows:

- **PM-Analog:** acPMAnalogConfigurationResetTotalCounters (Applicable only to Analog devices)
- **PM-Control:** acPMControlConfigurationResetTotalCounters
- **PM-Media:** acPMMediaConfigurationResetTotalCounters
- **PM-PSTN:** acPMPSTNConfigurationResetTotalCounters (Applicable only to Digital PSTN devices)
- **PM-System:** acPMSystemConfigurationResetTotalCounters

4.6.2 Performance Monitoring Parameters

The device's SNMP MIB PM parameters are listed in the subsequent subsections.

4.6.2.1 DS3 Performance Monitoring

Note: These PM parameters are applicable only to Mediant 3000 with the TP-6310 blade.

DS3 Performance Monitoring

MIB Name	Gauge/Counter	Description
dsx3IntervalPESs	Gauge	The counter associated with the number of P-bit Errored Seconds. EMS Parameter Name: DS3 PESs
dsx3IntervalPSESs	Gauge	The counter associated with the number of P-bit Severely Errored Seconds. EMS Parameter Name: DS3 PSESs
dsx3IntervalUASs	Gauge	The counter associated with the number of Unavailable Seconds. This object may decrease if the occurrence of unavailable seconds occurs across an interval boundary. EMS Parameter Name: DS3 UASs
dsx3IntervalLCVs	Gauge	The counter associated with the number of Line Coding Violations. EMS Parameter Name: DS3 LCVs

MIB Name	Gauge/Counter	Description
DS3 PCVs	Gauge	The counter associated with the number of P-bit Coding Violations. EMS Parameter Name: dsx3IntervalPCVs
dsx3IntervalLESs	Gauge	The number of Line Errored Seconds (BPVs or illegal zero sequences). EMS Parameter Name: DS3 LESs
dsx3IntervalCCVs	Gauge	The number of C-bit Coding Violations. EMS Parameter Name: DS3 CCVs
dsx3IntervalCESs	Gauge	The number of C-bit Errored Seconds. EMS Parameter Name: DS3 CESs
dsx3IntervalCSESs	Gauge	The number of C-bit Severely Errored Seconds. EMS Parameter Name: DS3 CSESs
dsx3CurrentPESs	-	The counter associated with the number of P-bit Errored Seconds.
dsx3CurrentPSESs	-	The counter associated with the number of P-bit Severely Errored Seconds.
dsx3CurrentUASs	-	The counter associated with the number of Unavailable Seconds.
dsx3CurrentLCVs	-	The counter associated with the number of Line Coding Violations.
dsx3CurrentPCVs	-	The counter associated with the number of P-bit Coding Violations.
dsx3CurrentLESs	-	The number of Line Errored Seconds.
dsx3CurrentCCVs	-	The number of C-bit Coding Violations.
dsx3CurrentCESs	-	The number of C-bit Errored Seconds.
dsx3CurrentCSESs	-	The number of C-bit Severely Errored Seconds.

4.6.2.2 Fiber Group Performance Monitoring

Note: These PM parameters are applicable only to Mediant 3000 with the TP-6310 blade.

Fiber Group Performance Monitoring

MIB Name	Gauge/Counter	Description
sonetSectionCurrentESs	Gauge	The counter associated with the number of Errored Seconds encountered by a SONET/SDH Section in the current 15 minute interval. EMS Parameter Name: Section ESs
sonetSectionCurrentSESs	Gauge	The counter associated with the number of Severely Errored Seconds encountered by a SONET/SDH Section in the current 15 minute interval. EMS Parameter Name: Section SESs
sonetSectionCurrentCVs	Gauge	The counter associated with the number of Coding Violations encountered by a SONET/SDH Section in the current 15 minute interval. EMS Parameter Name: Section CVs
sonetLineCurrentESs	Gauge	The counter associated with the number of Errored Seconds encountered by a SONET/SDH Line in the current 15 minute interval. EMS Parameter Name: Line ESs
sonetLineCurrentSESs	Gauge	The counter associated with the number of Severely Errored Seconds encountered by a SONET/SDH Line in the current 15 minute interval. EMS Parameter Name: Line SESs
sonetLineCurrentCVs	Gauge	The counter associated with the number of Coding Violations encountered by a SONET/SDH Line in the current 15 minute interval. EMS Parameter Name: Line CVs
sonetLineCurrentUASs	Gauge	The counter associated with the number of Unavailable Seconds encountered by a SONET/SDH Line in the current 15 minute interval. EMS Parameter Name: Line UASs
sonetPathCurrentESs	Gauge	The counter associated with the number of Errored Seconds encountered by a SONET/SDH Path in the current 15 minute interval. EMS Parameter Name: Path ESs
sonetPathCurrentSESs	Gauge	The counter associated with the number of Severely Errored Seconds encountered by a SONET/SDH Path in the current 15 minute interval. EMS Parameter Name: Path SESs

MIB Name	Gauge/Counter	Description
sonetPathCurrentCVs	Gauge	The counter associated with the number of Coding Violations encountered by a SONET/SDH Path in the current 15 minute interval. EMS Parameter Name: Path CVs
sonetPathCurrentUASs	Gauge	The counter associated with the number of Unavailable Seconds encountered by a Path in the current 15 minute interval. EMS Parameter Name: Path UASs

4.6.2.3 System IP Performance Monitoring

System IP Performance Monitoring

MIB Name	Gauge/Counter	Description
acPMNetUtilKBytesVolumeTx	Counter	Counts the total number of outgoing Kbytes (1000 bytes) from the interface during the last interval. EMS Parameter Name: Number of Outgoing KBytes
acPMNetUtilKBytesVolumeRx	Counter	Counts the total number of Kbytes (1000 bytes) received on the interface, including those received in error, during the last interval. EMS Parameter Name: Number of Incoming KBytes
acPMNetUtilPacketsVolumeTx	Counter	Counts the total number of outgoing Packets from the interface during the last interval. EMS Parameter Name: Number of Outgoing Pkts
acPMNetUtilPacketsVolumeRx	Counter	Counts the total number of Packets received on the interface, including those received in error, during the last interval. EMS Parameter Name: Number of Incoming Pkts
acPMNetUtilDiscardedPacketsVal	Counter	Counts the total number of malformed IP Packets received on the interface during the last interval. These are packets which are corrupted or discarded due to errors in their IP headers, including bad checksums, version number mismatch, other format errors, time-to-live exceeded, errors discovered in processing their IP options, etc. EMS Parameter Name: Number of Incoming Discarded Pkts

MIB Name	Gauge/Counter	Description
acPMNetUtilKBytesTotalTx	Gauge	This attribute counts the Current total number of outgoing Kbytes (1000 bytes) from the interface, so far from the beginning of the current collection interval as indicated by time Interval. EMS Parameter Name: Number of Outgoing KBytes
acPMNetUtilKBytesTotalRx	Gauge	This attribute counts the total number of Kbytes (1000 bytes) received on the interface, including those received in error, so far from the beginning of the current collection interval as indicated by time Interval. EMS Parameter Name: Number of Incoming KBytes
acPMNetUtilPacketsTotalTx	Gauge	This attribute counts the Current total number of outgoing Packets from the interface, so far from the beginning of the current collection interval as indicated by time Interval. EMS Parameter Name: Number of Outgoing Pkts
acPMNetUtilPacketsTotalRx	Gauge	This attribute counts the Current total number of Packets received on the interface, including those received in error, so far from the beginning of the current collection interval as indicated by time Interval. EMS Parameter Name: Number of Incoming Pkts
acPMNetUtilDiscardedPacketsTotal	Gauge	This attribute counts the Current total number of malformed IP Packets received on the interface from the beginning of the current collection interval. These are packets which are corrupted or discarded due to errors in their IP headers, including bad checksums, version number mismatch, other format errors, time-to-live exceeded, errors discovered in processing their IP options, etc. EMS Parameter Name: Number of Incoming Discarded Pkts

4.6.2.4 VoP Call Statistics Performance Monitoring

Note: These PM parameters are not applicable to the MediaPack series.

VoP Call Statistics Performance Monitoring

MIB Name	Gauge / Counter	Description
acPMActiveContextCountAverage	Gauge	Indicates the average number of voice calls connected on the gateway since the last clear. EMS Parameter Name: Num of Active Contexts Avg
acPMActiveContextCountMin	Gauge	Indicates the minimum number of voice calls connected on the gateway since the last clear. EMS Parameter Name: Num of Active Contexts Min
acPMActiveContextCountMax	Gauge	Indicates the maximum number of voice calls connected on the gateway since the last clear. EMS Parameter Name: Num of Active Contexts Max
acPMChannelsPerCoderAverageG711	Gauge	Indicates the average number of G.711 calls present on the TPM. EMS Parameter Name: G711 Active Calls Avg
acPMChannelsPerCoderAverageG723	Gauge	Indicates the average number of G.723 calls present on the TPM. This attribute is only displayed if the G.723 Codec is provisioned on the DSP template. EMS Parameter Name: G723 Active Calls Avg
acPMChannelsPerCoderAverageG728	Gauge	Indicates the average number of G.728 calls present on the TPM. This attribute is only displayed if the G.728 Codec is provisioned on the DSP template. EMS Parameter Name: G728 Active Calls Avg
acPMChannelsPerCoderAverageG729a	Gauge	Indicates the average number of G.729a calls present on the TPM. This attribute is only displayed if the G.729a Codec is provisioned on the DSP. EMS Parameter Name: G729a Active Calls Avg
acPMChannelsPerCoderAverageG729e	Gauge	Indicates the average number of G.729e calls present on the TPM. This attribute is only displayed if the G.729e Codec is provisioned on the DSP template. EMS Parameter Name: G729e Active Calls Avg
acPMChannelsPerCoderAverageAMR	Gauge	Indicates the average number of AMR calls present on the TPM. This attribute is only displayed if the AMR Codec is provisioned on the DSP template. EMS Parameter Name: AMR Active Calls Avg

MIB Name	Gauge / Counter	Description
acPMModuleRTPPacketLossRxMax	Gauge	Indicates the Max Rx RTP Packet loss (reported by RTCP) per TPM, up to this point in time during the collection interval, as indicated by the time Interval. EMS Parameter Name: Rx RTP Packet Loss Max
acPMModuleRTPPacketLossTxMax	Gauge	Indicates the Max Tx RTP Packet loss (reported by RTCP) per TPM, up to this point in time during the collection interval, as indicated by the time Interval. EMS Parameter Name: Tx RTP Packet Loss Max
acPMModulePacketDelayAverage	Gauge	Indicates the average RTP packets delay per TPM, up to this point in time during the collection interval, as indicated by the time Interval. EMS Parameter Name: RTP delay Average
acPMModulePacketDelayMax	Gauge	Indicates the maximum RTP packets delay per TPM, up to this point in time during the collection interval, as indicated by the time Interval. EMS Parameter Name: RTP delay Max
acPMModulePacketDelayMin	Gauge	Indicates the minimum RTP packets delay per TPM, up to this point in time during the collection interval, as indicated by the time Interval. EMS Parameter Name: RTP delay Min
acPMModulePacketJitterAverage	Gauge	Indicates the average RTP packets jitter per TPM, up to this point in time during the collection interval, as indicated by the time Interval. EMS Parameter Name: RTP jitter Average
acPMModulePacketJitterMin	Gauge	Indicates the minimum RTP packets jitter per TPM, up to this point in time during the collection interval, as indicated by the time Interval. EMS Parameter Name: RTP jitter Min
acPMModulePacketJitterMax	Gauge	Indicates the maximum RTP packets jitter per TPM, up to this point in time during the collection interval, as indicated by the time Interval. EMS Parameter Name: RTP jitter Max
acPMModuleRTPBytesRxMax	Gauge	Indicates the Max Tx RTP Bytes per TPM, up to this point in time during the collection interval, as indicated by the time Interval. EMS Parameter Name: Rx RTP Bytes Max
acPMModuleRTPBytesTxMax	Gauge	Indicates the Max Rx RTP Bytes per TPM, up to this point in time during the collection interval, as indicated by the time Interval. EMS Parameter Name: Tx RTP Bytes Max

MIB Name	Gauge / Counter	Description
acPMModuleRTPPacketsRxMax	Gauge	Indicates the Max Rx RTP Packets per TPM, up to this point in time during the collection interval, as indicated by the time Interval. EMS Parameter Name: Rx RTP Packets Max
acPMModuleRTPPacketsTxMax	Gauge	Indicates the Max Tx RTP Packets per TPM, up to this point in time during the collection interval, as indicated by the time Interval. EMS Parameter Name: Tx RTP Packets Max
acPMActiveContextCountVal	Gauge	Indicates the current number of voice calls connected on the box since last clear. EMS Parameter Name: Num of Active Contexts
acPMChannelsPerCoderValG711	Gauge	This attribute indicates the current number of G711 calls present on the TPM. EMS Parameter Name: G711 Active Calls
acPMChannelsPerCoderValG723	Gauge	This attribute indicates the current number of G723 calls present on the TPM.This attribute is only displayed if the G723 Codec is provisioned on the DSP template. EMS Parameter Name: G723 Active Calls
acPMChannelsPerCoderValG728	Gauge	This attribute indicates the current number of G728 calls present on the TPM.This attribute is only displayed if the G728 Codec is provisioned on the DSP template. EMS Parameter Name: G728 Active Calls
acPMChannelsPerCoderValG729a	Gauge	This attribute indicates the current number of G729a calls present on the TPM.This attribute is only displayed if the G729a Codec is provisioned on the DSP. EMS Parameter Name: G729a Active Calls
acPMChannelsPerCoderValG729e	Gauge	This attribute indicates the current number of G729e calls present on the TPM.This attribute is only displayed if the G729e Codec is provisioned on the DSP template. EMS Parameter Name: G729e Active Calls
acPMChannelsPerCoderValAMR	Gauge	This attribute indicates the current number of AMR calls present on the TPM.This attribute is only displayed if the AMR Codec is provisioned on the DSP template. EMS Parameter Name: AMR Active Calls
acPMModuleRTPPacketLossRxTotal	Gauge	The total number of RTP packet loss reported by RTCP since last reset. EMS Parameter Name: Rx Packet Loss current
acPMModuleRTPPacketLossTxTotal	Gauge	The total number of RTP packet loss reported by RTCP since last reset. EMS Parameter Name: Tx Packets Loss current
acPMModuleRTPPacketsRxTotal	Gauge	The total number of packets recieved since last reset. EMS Parameter Name: Rx Packets Current

MIB Name	Gauge / Counter	Description
acPMModuleRTPPacketsTxTotal	Gauge	The total number of RTP packets transmited since last reset. EMS Parameter Name: Rx Packets Current

4.6.2.5 Common Control Performance Monitoring

Note: These PM parameters are not applicable to the MediaPack series.

Common Control Performance Monitoring

MIB Name	Gauge/Counter	Description
acPMCPConnectionLifetimeAverage	Counter	Indicates the Connection lifetime, in seconds. EMS Parameter Name: Lifetime in seconds Avg
acPMCPConnectionLifetimeMin	Counter	Indicates the Connection lifetime, in seconds. EMS Parameter Name: Lifetime in seconds Min
acPMCPConnectionLifetimeMax	Counter	Indicates the Connection lifetime, in seconds. EMS Parameter Name: Lifetime in seconds Max
acPMCPCommandCounterValRx	Counter	Indicates the MGC response counters. EMS Parameter Name: MGC response counters
acPMCPCommandCounterValTx	Counter	Indicates the MGC command counters. EMS Parameter Name: MGC command counters
acPMCPRetransmissionCountValRx	Counter	Counts the number of incoming retransmissions. EMS Parameter Name: MGC Rx retransmissions
acPMCPRetransmissionCountValTx	Counter	Counts the number of transactions retransmissions sent from the board. EMS Parameter Name: MGC Tx retransmissions
acPMCPCallAttemptsPerSecAverage	Counter	Average of call attempts (successful and unsuccessful) per second, during last interval. EMS Parameter Name: Call Attempts Per Sec Average

MIB Name	Gauge/Counter	Description
acPMCPCallAttemptsPerSecMax	Counter	Maximum of call attempts (successful and unsuccessful) per second, during last interval. EMS Parameter Name: Call Attempts Per Sec Max
acPMCPCallAttemptsPerSecMin	Counter	Minimum of call attempts (successful and unsuccessful) per second, during last interval. EMS Parameter Name: Call Attempts Per Sec Min
acPMCPConnectionLifetimeVolume	Counter	The Connection lifetime in seconds. EMS Parameter Name: Lifetime in seconds
acPMCPCommandCounterTotalTx	Gauge	MGC command counters. EMS Parameter Name: MGC Tx command counters
acPMCPCommandCounterTotalRx	Gauge	MGC response counters. EMS Parameter Name: MGC Rx command counters
acPMCPRetransmissionCountTotalTx	Gauge	Number of transactions retransmissions sent from the board. EMS Parameter Name: MGC Tx retransmissions
acPMCPRetransmissionCountTotalRx	Gauge	Number of incoming retransmissions. EMS Parameter Name: MGC Rx retransmissions
acPMCPCallAttemptsPerSecVal	Gauge	Number of Call attempts (successful and unsuccessful) per second, during current interval. EMS Parameter Name: Call Attempts Per Sec

4.6.2.6 SIP IP-to-Tel Performance Monitoring

Note: These PM parameters are not applicable to Mediant 4000.

SIP IP-to-Tel Performance Monitoring

MIB Name	Gauge/Counter	Description
acPMSIPAttemptedCallsVal	Counter	Indicates the number of attempted calls for IP to Tel direction, during last interval. EMS Parameter Name: IP to Tel Number of Call Attempts

MIB Name	Gauge/Counter	Description
acPMSIPEstablishedCallsVal	Counter	Indicates the number of established calls for IP to Tel direction, during last interval. EMS Parameter Name: IP to Tel Number of Established Calls
acPMSIPBusyCallsVal	Counter	Indicates the number of calls that failed as a result of a busy line for IP to Tel direction, during last interval. EMS Parameter Name: IP to Tel Number of Calls Terminated due to a Busy Line
acPMSIPNoAnswerCallsVal	Counter	Indicates the number of calls that weren't answered for IP to Tel direction, during last interval. EMS Parameter Name: IP to Tel Number of Calls Terminated due to No Answer
acPMSIPForwardedCallsVal	Counter	Indicates the number of calls that were terminated due to a call forward for IP to Tel direction, during last interval. EMS Parameter Name: IP to Tel Number of Calls Terminated due to Forward
acPMSIPNoRouteCallsVal	Counter	Indicates the number of calls whose destinations weren't found for IP to Tel direction, during last interval. EMS Parameter Name: IP to Tel Number of Failed Calls due to No Route
acPMSIPNoMatchCallsVal	Counter	Indicates the number of calls that failed due to mismatched media server capabilities for IP to Tel direction, during last interval. EMS Parameter Name: IP to Tel Number of Failed Calls due to No Matched Capabilities
acPMSIPNoResourcesCallsVal	Counter	Indicates the number of calls that failed due to unavailable resources or a media server lock for IP to Tel direction, during last interval. EMS Parameter Name: IP to Tel Number of Failed Calls due to No Resources
acPMSIPFailCallsVal	Counter	This counter is incremented as a result of calls that fail due to reasons not covered by the other counters for IP to Tel direction, during last interval. EMS Parameter Name: IP to Tel Number of Failed Calls due to Other reasons
acPMSIPCallDurationAverage	Gauge	Indicates the average call duration of established calls for IP to Tel direction, during last interval. EMS Parameter Name: IP to Tel Average Call Duration [sec]
IP2TelTrunkGroupEstablishedCalls	Gauge	Indicates the current number of established calls pertaining to a Trunk Group for IP to Tel direction.

MIB Name	Gauge/Counter	Description
IP2TelTrunkEstablishedCalls	Gauge	Indicates the current number of established calls pertaining to a trunk for IP to Tel direction.

4.6.2.7 SIP Tel-to-IP Performance Monitoring

Note: These PM parameters are not applicable to Mediant 4000.

SIP Tel-to-IP Performance Monitoring

MIB Name	Gauge/Counter	Description
acPMSIPAttemptedCallsVal	Counter	Indicates the number of attempted calls for Tel to IP direction, during last interval. EMS Parameter Name: Tel to IP Number of Call Attempts
acPMSIPEstablishedCallsVal	Counter	Indicates the number of established calls for Tel to IP direction, during last interval. EMS Parameter Name: Tel to IP Number of Established Calls
acPMSIPBusyCallsVal	Counter	Indicates the number of calls that failed as a result of a busy line for Tel to IP direction, during last interval. EMS Parameter Name: Tel to IP Number of Calls Terminated due to a Busy Line
acPMSIPNoAnswerCallsVal	Counter	Indicates the number of calls that weren't answered for Tel to IP direction, during last interval. EMS Parameter Name: Tel to IP Number of Calls Terminated due to No Answer
acPMSIPForwardedCallsVal	Counter	Indicates the number of calls that were terminated due to a call forward for Tel to IP direction, during last interval. EMS Parameter Name: Tel to IP Number of Calls Terminated due to Forward
acPMSIPNoRouteCallsVal	Counter	Indicates the number of calls whose destinations weren't found for Tel to IP direction, during last interval. EMS Parameter Name: Tel to IP Number of Failed Calls due to No Route
acPMSIPNoMatchCallsVal	Counter	Indicates the number of calls that failed due to mismatched media server capabilities for Tel to IP direction, during last interval. EMS Parameter Name: Tel to IP Number of Failed Calls due to No Matched Capabilities

MIB Name	Gauge/Counter	Description
acPMSIPNoResourcesCallsVal	Counter	Indicates the number of calls that failed due to unavailable resources or a media server lock for Tel to IP direction, during last interval. EMS Parameter Name: Tel to IP Number of Failed Calls due to No Resources
acPMSIPFailCallsVal	Counter	This counter is incremented as a result of calls that fail due to reasons not covered by the other counters for Tel to IP direction, during last interval. EMS Parameter Name: Tel to IP Number of Failed Calls due to Other reasons
acPMSIPCallDurationAverage	Gauge	Indicates the average call duration of established calls for Tel to IP direction, during last interval. EMS Parameter Name: Tel to IP Average Call Duration [sec]
Tel2IPTrunkEstablishedCalls	Gauge	Indicates the current number of established calls pertaining to a trunk for Tel to IP direction.
Tel2IPTrunkGroupEstablishedCalls	Gauge	Indicates the current number of established calls pertaining to a Trunk Group for Tel to IP direction.

4.6.2.8 Media Realms Statistics Performance Monitoring

Media Realm Statistics Performance Monitoring

MIB Name	Gauge/Counter	Description
PM_RTPRealmBytesTx	Gauge	RTP bytes received in RTCP data, per realm
PM_RTPRealmBytesRx	Gauge	RTP bytes received in RTCP data, per realm
PM_RTPRealmPacketsTx	Gauge	RTP packets sent in RTCP data, per realm
PM_RTPRealmPacketsRx	Gauge	RTP packets received in RTCP data, per realm
PM_RTPRealmPacketLossRx	Gauge	RTP packet loss reported in outgoing RTCP data, per realm
PM_RTPRealmPacketLossTx	Gauge	RTP packet loss reported in incoming RTCP data, per realm
PM_MediaRealmBytesTx	Gauge	Media bytes received in RTCP data, per realm

MIB Name	Gauge/Counter	Description
PM_MediaRealmBytesRx	Gauge	Media bytes received in RTCP data, per realm
PM_MediaRealmPacketsTx	Gauge	Media packets sent in RTCP data, per realm
PM_MediaRealmPacketsRx	Gauge	Media packets received in RTCP data, per realm
PM_VERealmPacketDelay	Gauge	Packet delay in RTCP data, per realm
PM_VERealmPacketJitter	Gauge	Packet jitter in RTCP data, per realm
PM_RealmMOS	Gauge	MOS quality in RTCP-XR data, per realm
PM_MediaRealmBwRx	Gauge	Average bandwidth for Rx bytes, per realm
PM_MediaRealmBwTx	Gauge	Average bandwidth for Tx bytes, per realm

4.6.2.9 SBC Admission Control Performance Monitoring

The device now provides SNMP performance monitoring for the SBC Admission Control feature. The performance monitoring is performed per the following:

- SRD/IP Group
- Incoming, outgoing, or both
- SIP request types - INVITE, SUBSCRIBE, OTHER, or ALL

The performance monitoring is provided by the acGateway MIB.

Note: This section is applicable only to the MSBG series, Mediant 3000, and Mediant 4000.

SBC Call Admision Control Performance Monitoring

MIB Name	Gauge / Counter	Description
acPMSIPSRDDialogsTable	Counter	Counter for all dialogs currently being handled by the SBC per SRD
acPMSIPSRDInviteDialogsTable	Counter	Counter of all calls (initiated by SIP:INVITE) currently being handled by the SBC per SRD
acPMSIPSRDSubscribeDialogsTable	Counter	Counter of all SUBSCRIBE dialogs (initiated by SIP:SUBSCRIBE) currently being handled by the SBC per SRD
acPMSIPSRDOtherDialogsTable	Counter	Counter of all dialogs other than INVITE and SUBSCRIBE (initiated by SIP:REGISTER) currently being handled by the SBC per SRD
acPMSIPIPGroupDialogsTable	Counter	Counter for all dialogs currently being handled by the SBC per IP Group
acPMSIPIPGroupInviteDialogsTable	Counter	Counter of all calls (initiated by SIP:INVITE) currently being handled by the SBC per IP Group

MIB Name	Gauge / Counter	Description
acPMSIPIPGroupSubscribeDialogsTable	Counter	Counter of all SUBSCRIBE dialogs (initiated by SIP:SUBSCRIBE) currently being handled by the SBC per IP Group
acPMSIPIPGroupOtherDialogsTable	Counter	Counter of all other dialogs other than INVITE and SUBSCRIBE (initiated by SIP:REGISTER) currently being handled by the SBC per IP Group

4.6.2.10 Trunk Statistics Performance Monitoring

Note: These PM parameters are applicable only to Digital PSTN devices.

Trunk Statistics Performance Monitoring

MIB Name	Gauge/Counter	Description
acPMTrunkUtilizationAverage	Gauge	Indicates the Average of simultaneously busy DS0 channels on this Trunk up to this point in time during the collection interval, as indicated by the Time Interval. A busy channel is when the Physical DS0 Termination isn't in Null context or OOS. A Trunk is either E1 or T1. EMS Parameter Name: Trunk utilization Avg
acPMTrunkUtilizationMin	Gauge	Indicates the Minimum of simultaneously busy DS0 channels on this Trunk up to this point in time during the collection interval, as indicated by the Time Interval. A busy channel is when the Physical DS0 Termination isn't in Null context or OOS. A Trunk is either E1 or T1. EMS Parameter Name: Trunk utilization Min
acPMTrunkUtilizationMax	Gauge	Indicates the Maximum of simultaneously busy DS0 channels on this Trunk up to this point in time during the collection interval, as indicated by the Time Interval. A busy channel is when the Physical DS0 Termination isn't in Null context or OOS. A Trunk is either E1 or T1. EMS Parameter Name: Trunk utilization Max
dsx1IntervalESs	Gauge	Indicates the number of Errored Seconds. EMS Parameter Name: Trunk Errored Seconds

MIB Name	Gauge/Counter	Description
dsx1IntervalCSSs	Gauge	Indicates the number of Controlled Slip Seconds. EMS Parameter Name: Trunk Controlled Slip Seconds
dsx1IntervalPCVs	Gauge	Indicates the number of Path Coding Violations. EMS Parameter Name: Trunk Path Coding Violations
dsx1IntervalBESs	Gauge	Indicates the number of Bursty Errored Seconds. EMS Parameter Name: Trunk Bursty Errored Seconds
acPMTrunkUtilizationVal	Gauge	This attribute indicates the Current simultaneous busy DS0 channels on this Trunk. A busy channel is when the Physical DS0 Termination isn't in Null context or OOS. A Trunk is either E1 or T1. EMS Parameter Name: Trunk utilization
acPMPSTNTrunkActivitySecondsTotal	Gauge	This attribute amount of call duration per timeslot and E1 since last clear. EMS Parameter Name: Trunk Calls Duration
dsx1TotalESs	Gauge	This attribute indicates amount of Errored Seconds encountered by a DS1 interface in the previous 24 hour interval. Invalid 15 minute intervals count as 0. EMS Parameter Name: Trunk Errored Seconds
dsx1TotalCSSs	Gauge	This attribute indicates amount of Controlled Slip Seconds encountered by a DS1 interface in the previous 24 hour interval. Invalid 15 minute intervals count as 0. EMS Parameter Name: Trunk Controlled Slip Seconds
dsx1TotalPCVs	Gauge	This attribute indicates amount of Path Coding Violations encountered by a DS1 interface in the previous 24 hour interval. Invalid 15 minute intervals count as 0. EMS Parameter Name: Trunk Path Coding Violations
dsx1TotalBESs	Gauge	This attribute indicates amount of Bursty Errored Seconds encountered by a DS1 interface in the previous 24 hour interval. Invalid 15 minute intervals count as 0. EMS Parameter Name: Trunk Bursty Errored Seconds

4.7 TrunkPack-VoP Series Supported MIBs

The device contains an embedded SNMP agent supporting the listed MIBs below. A description in HTML format for all supported MIBs can be found in the MIBs directory in the release package.

- **The Standard MIB (MIB-2):** The various SNMP values in the standard MIB are defined in RFC 1213. The standard MIB includes various objects to measure and monitor IP activity, TCP activity, UDP activity, IP routes, TCP connections, interfaces, and general system description.
    - The standard icmpStatsTable and icmpMsgStatsTable under MIB-2 support ICMP statistics for both IPv4 and IPv6.
    - The inetCidrRouteTable (from the standard IP-FORWARD-MIB) supports both IPv4 and IPv6.

Note: For Mediant 3000/TP-6310 and Mediant 2000/TP-1610: In the ipCidrRouteIfIndex, the IF MIB indices are not referenced. Instead, the index used is related to one of the IP interfaces in the blade: (1) OAMP, (2) Media, and (3) Control. When there is only one interface, the only index is OAMP (1). Refer to the device's *User's Manual*.

- **System MIB (under MIB-2):** The standard system group: sysDescr, sysObjectID, sysUpTime, sysContact, sysName, sysLocation, and sysServices. You can replace the value of sysObjectID.0 with variable value using the *ini* file parameter that calls SNMPSysOid. This parameter is polled during the startup and overwrites the standard sysObjectID. SNMPSysName is an administratively assigned name for this managed node. By convention, this is the node's fully-qualified domain name. If the name is unknown, the value is the zero-length string.
- **Host Resources MIB (RFC 2790):** The Host Resources MIB is used for managing host systems. The term host is any computer that communicates with other similar computers connected to the Internet and that is directly used by one or more human beings. The following are the Host Resources MIB objects:
    - hrSystem group
    - hrStorage group (basic only)
    - hrDevice group (CPU, RAM, Flash - basic only)
    - hrSWRunPerf (basic only)
    - hrSWInstalled (OS only)

  Applicable Products: All devices.
- **RTP MIB:** The RTP MIB is supported according to RFC 2959. It contains objects relevant to the RTP streams generated and terminated by the device and to the RTCP information related to these streams.

Note: The inverse tables are not supported.

- **Notification Log MIB:** Standard MIB (RFC 3014 - iso.org.dod.internet.mgmt.mib-2) supported for implementation of Carrier Grade Alarms.

- **Alarm MIB:** IETF MIB (RFC 3877) supported as part of the implementation of Carrier Grade Alarms.
- **SNMP Target MIB**: (RFC 2273) allows for configuration of trap destinations and trusted managers.
- **SNMP MIB:** (RFC 3418) allows support for the coldStart and authenticationFailure traps.
- **SNMP Framework MIB:** (RFC 3411).
- **SNMP Usm MIB:** (RFC 3414) implements the user-based Security Model.
- **SNMP Vacm MIB:** (RFC 3415) implements the view-based Access Control Model.
- **SNMP Community MIB:** (RFC 3584) implements community string management.
- **ipForward MIB:** (RFC 2096) - fully supported.
- **RTCP-XR:** (RFC) implements the following partial support (applicable to all except MP):
    - The rtcpXrCallQualityTable is fully supported.
    - In the rtcpXrHistoryTable, support of the RCQ objects is provided only with no more than 3 intervals, 15 minutes long each.
    - Supports the rtcpXrVoipThresholdViolation trap.
- **ds1 MIB:** supports the following (Applicable only to Digital PSTN devices):
    - dsx1ConfigTable: partially supports the following objects with SET and GET applied:
        - dsx1LineCoding
        - dsx1LoopbackConfig
        - dsx1LineStatusChangeTrapEnable
        - dsx1CircuitIdentifier

  All other objects in this table support GET only.

    - dsx1CurrentTable
    - dsx1IntervalTable
    - dsx1TotalTable
    - dsx1LineStatusChange trap
- **ds3 MIB:** (RFC 3896) supports the following (Applicable only to the Mediant 3000):
    - dsx3ConfigTable: refer to the supplied MIB version for limits on specific objects. The table includes the following objects:
        - TimerElapsed
        - ValidIntervals
    - dsx3LineStatusChange: The following tables (RFC 2496) are supported:
        - dsx3CurrentTable
        - dsx3IntervalTable
        - dsx3TotalTable

Proprietary MIB objects that are related to the SONET/SDH configuration (applicable only to Mediant 3000 with TP-6310):

- **In the acSystem MIB**:
    - acSysTransmissionType: sets the transmission type to optical or DS3 (T3).
- **SONET MIB:** (RFC 3592) implements the following partial support:
    - In the SonetMediumTable, the following objects are supported:
        - SonetMediumType
        - SonetMediumLineCoding
        - SonetMediumLineType
        - SonetMediumCircuitIdentifier
        - sonetMediumLoopbackConfig
    - In the SonetSectionCurrentTable, the following objects are supported:
        - lsonetSectionCurrentStatus
        - sonetSectionCurrentESs
        - sonetSectionCurrentSESs
        - sonetSectionCurrentSEFSs
        - sonetSectionCurrentCVs
    - In the SonetLineCurrentTable, the following objects are supported:
        - sonetLineCurrentStatus
        - sonetLineCurrentESs
        - sonetLineCurrentSESs
        - sonetLineCurrentCVs
        - sonetLineCurrentUASs
    - sonetSectionIntervalTable
    - sonetLineIntervalTable
    - sonetPathCurrentTable
    - sonetPathIntervalTable
- **Traps (refer AcBoard MIB for additional details):**
    - SONET (applicable only to Mediant 3000 with TP-6310):
        - acSonetSectionLOFAlarm
        - acSonetSectionLOSAlarm
        - acSonetLineAISAlarm
        - acSonetLineRDIAlarm
        - acSonetPathSTSLOPAlarm
        - acSonetPathSTSAISAlarm
        - acSonetPathSTSRDIAlarm
        - acSonetPathUnequippedAlarm
        - acSonetPathSignalLabelMismatchAlarm

    - DS3 (applicable only to Mediant 3000 with TP-6310):
        - acDS3RAIAlarm - DS3 RAI alarm
        - acDS3AISAlarm - DS3 AIS alarm
        - acDS3LOFAlarm - DS3 LOF alarm
        - acDS3LOSAlarm - DS3 LOS alarm
    - acSonetIfHwFailureAlarm

- **In the acPSTN MIB:**
    - acSonetSDHTable: currently has one entry (acSonetSDHFbrGrpMappingType) for selecting a low path mapping type. Relevant only for PSTN applications. (Refer to the MIB for more details.)
- **In the acSystem MIB:**
    - acSysTransmissionType: sets the transmission type to optical or DS3 (T3).

In addition to the standard MIBs, the complete product series contains proprietary MIBs:

- **AC-TYPES MIB:** lists the known types defined by the complete product series. This is referred to by the sysObjectID object in the MIB-II.
- The AcBoard MIB includes the following group: **acTrap**

Note: The AcBoard MIB is being phased out.

Each proprietary MIB contains a Configuration subtree for configuring the related parameters. In some, there also are Status and Action subtrees.

- **AcAnalog MIB** (Applicable only to Analog devices)
- **acControl MIB**
- **acMedia MIB**
- **acSystem MIB**
- **acSysInterfaceStatusTable:** supports the networking multiple interfaces feature status. This table reflects all the device's active interfaces. The lines indices consist of both the Entry Index and the Type Index. The table contains the following columns:
    - Entry Index - related Interface index in the interface configuration table (if the table is empty,i.e., there is only single IP address, the index appears with 0)
    - Type Index - 1 for IP Address and 2 for IPv6 Link-Local Address
    - Application Types - type assigned to the interface
    - Status Mode - interface configuration mode
    - IP Address - IP address (either IPv4 or IPv6) for this interface
    - Prefix Length - number of ‘1’ bits in this interface’s net mask
    - Gateway - default gateway
    - Vlan ID - VLAN ID of this interface
    - Name - interface’s name

  - Primary DNS Server IP Address - IP address of primary DNS server for this interface
  - Secondary DNS Server IP Address - IP address of secondary DNS server for this interface

- **acSysEthernetStatusTable** - Ethernet relevant information. (Applicable only to Mediant 3000 with TP-8410 Blade)
- **acSysModuleTable** (Applicable only to 8410 Blade Series)
- **acIPMediaChannelsresourcesTable** - IPMedia channels information such as Module ID and DSP Channels Reserved (Applicable only to Mediant 1000)
- **acPSTN MIB** (Applicable only to Digital PSTN devices)
- **acGateway MIB:** This proprietary MIB contains objects related to configuration of the SIP device. This MIB complements the other proprietary MIBs.

  The acGateway MIB includes the following groups:

  - **Common**: parameters common to both SIP and H.323.
  - **SIP**: SIP only parameters.

- **AcAlarm:** This is a proprietary carrier-grade alarm MIB. It is a simpler implementation of the notificationLogMIB and the IETF suggested alarmMIB (both also supported in all devices).

  The acAlarm MIB has the following groups:

  - **ActiveAlarm**: straight forward (single indexed) table listing all currently active Alarms together with their bindings (the Alarm bindings are defined in acAlarm. acAlarmVarbinds and also in acBoard.acTrap. acBoardTrapDefinitions. oid_1_3_6_1_4_1_5003_9_10_1_21_2_0).
  - **acAlarmHistory**: straight forward (single indexed) table listing all recently raised Alarms together with their bindings (the Alarm bindings are defined in acAlarm. acAlarmVarbinds and also in acBoard.acTrap. acBoardTrapDefinitions. oid_1_3_6_1_4_1_5003_9_10_1_21_2_0).

  The table size can be altered via:

  notificationLogMIB.notificationLogMIBObjects.nlmConfig.nlmConfigGlobalEntryLimit or notificationLogMIB.notificationLogMIBObjects.nlmConfig.nlmConfigLogTable.nlmConfigLogEntry.nlmConfigLogEntryLimit.

  The table size (i.e., number of contained alarms) can be as follows:

  - Digital devices: Any value between 10 and 1,000 (default is 500)
  - MediaPack devices: Any value between 10 and 100 (default is 100)

Notes:

- A detailed explanation of each parameter can be viewed in the MIB Description field.
- A detailed description in HTML format of all MIBs can be found in the MIBs directory (included in the Release package).
- Not all groups in the MIB are implemented.
- MIB Objects that are marked as 'obsolete' are not implemented.
- When a parameter is Set to a new value via SNMP, the change may affect device functionality immediately or may require that the device be soft reset for the change to take effect. This depends on the parameter type.
- The current (updated) device configuration parameters are configured on the device provided the user doesn't load an *ini* file to the device after reset. Loading an *ini* file after reset overrides the updated parameters.

4.8 Traps

Full proprietary trap definitions and trap Varbinds are found in AcBoard MIB and AcAlarm MIB. For a detailed inventory of traps, see "SNMP Traps" on page 84.

Note: All traps are sent from the SNMP port (default 161).

The following proprietary traps are supported by the device:

Table 4-6: Proprietary Traps

Trap	Description
acBoardFatalError	Sent whenever a fatal device error occurs.
acBoardConfigurationError	Sent when the device's settings are invalid. The trap contains a message stating/detailing/explaining the invalid setting.
acBoardTemperatureAlarm	Sent when the device exceeds its temperature limits. **Note:** Applicable only to 2000 and 3000 Series devices.
acBoardEvResettingBoard	Sent after the device resets.
acBoardEvBoardstarted	Sent after the device is successfully restored and initialized following reset.
acFeatureKeyError	Sent to relay Feature Key errors etc.
acgwAdminStateChange	Sent when Graceful Shutdown commences and ends.
acBoardEthernetLinkAlarm	Sent when the Ethernet link(s) is down.
acBoardWanLinkAlarm	This alarm is raised when the WAN Link is down (and cleared when link is up again). **Note:** This alarm is applicable only to MSBG devices.

Trap	Description
acWirelessCellularModemAlarm	This alarm is raised when either the wireless modem is down or in backup mode, and cleared when modem is up. **Note:** This alarm is applicable only to Mediant 800 MSBG.
acActiveAlarmTableOverflow	Sent when an active alarm cannot be entered into the Active Alarm table because the table is full.
acAudioProvisioningAlarm	Sent if the device is unable to provision its audio. **Note:** Not applicable to MSBG.
acOperationalStateChange	Sent if the operational state of the node goes to disabled; cleared when the operational state of the node goes to enabled.
acNATTraversalAlarm	Sent when the NAT is placed in front of a device and is identified as a symmetric NAT. It is cleared when a non-symmetric NAT or no NAT replace the symmetric one. **Note:** Not applicable to MSBG.
acEnhancedBITStatus	Sent for the status of the BIT (Built In Test). The information in the trap contains blade hardware elements being tested and their status. The information is presented in the additional info fields. **Note:** Not applicable to MSBG.
acTMInconsistentRemoteAndLocalPLLStatus	Inconsistent Remote and Local PLL status. **Note:** Applicable only to Mediant 3000.
acTMReferenceStatus	Timing manager reference status. **Note:** Applicable only to Mediant 3000.
acTMReferenceChange	Timing manager reference change. **Note:** Applicable only to Mediant 3000.
acFanTrayAlarm	Sent when a fault occurs in the fan tray or a fan tray is missing. **Note:** Applicable only to Mediant 1000 and 3000 Series devices.
acPowerSupplyAlarm	Sent when a fault occurs in one of the power supply (PS) modules or a PS module is missing. **Note:** Applicable only to the 3000 Series devices.
acPEMAlarm	Sent when a fault occurs in one of the PEM modules or a PEM module is missing. **Note:** Applicable only to the 3000 Series devices.
acSAMissingAlarm	Sent when the SA module is missing or non operational. **Note:** Applicable only to the 3000 Series devices.
acUserInputAlarm	Sent when the input dry contact is short circuited; cleared when the circuit is reopened. **Note:** Applicable only to the 3000 Series devices.
acPerformanceMonitoringThresholdCrossing	Sent every time the threshold of a Performance Monitored object is crossed. The severity field is 'indeterminate' when the crossing is above the threshold and 'cleared' when it goes back under the threshold. The 'source' varbind in the trap indicates the object for which the threshold is being crossed.

Trap	Description
acNTPServerStatusAlarm	NTP server status alarm. Raised when the connection to the NTP server is lost. Cleared when the connection is reestablished. Unset time (as a result of no connection to NTP server) may result with functionality degradation and failure in device.
acDChannelStatus	Non-alarm trap sent at the establishment, re-establishment or release of LAPD link with its peer connection occurs. The trap is sent with one of the following textual descriptions: ▪ D-channel synchronized ▪ D-channel not-synchronized **Note:** Applicable only to the Digital PSTN devices.
acHTTPDownloadResult	Sent upon success or failure of the HTTP Download action.
acKeepAlive	Part of the NAT traversal mechanism. If the STUN application in the device detects a NAT, this trap is sent on a regular time laps - 9/10 of the acSysSTUNBindingLifeTime object. The AdditionalInfo1 varbind has the MAC address of the device.
acPowerOverEthernetStatus	This trap is sent when Power over Ethernet (PoE) for a specific port is disabled.
acMediaProcessOverloadAlarm	This alaram is raised upon overload of the device’s media processing and interfaces.
SIP Traps	
acBoardCallResourcesAlarm	Sent when no free channels are available.
acBoardControllerFailureAlarm	Sent when the Proxy is not found or registration fails. Internal routing table may be used for routing.
acBoardOverloadAlarm	Sent when there is an overload in one or some of the system's components.
acGWSASEmergencyModeAlarm	Sent by the Stand-Alone Survivability (SAS) application when switching from “Normal" mode to “Emergency" mode. This alarm is cleared once the SAS returns to “Normal" mode. **Note:** Applicable only to MediaPack, Mediant 800, Mediant 1000, and Mediant 2000.
GWAPP_TRAP_BUSYOUT_LINK	Sent when the physical network link is up or down ("BusyOut Trunk/Line n Link failure").
GWAPP_TRAP_BUSYOUT_CONNECTIVITY:	Sent when the connection to the Proxy is up or down ("BusyOut Trunk/Line n Connectivity Proxy failure").
GWAPP_TRAP_BUSYOUT_TDM_OVER_IP	Sent when a failure occurs in TDM over IP (transparent T1/E1 without signaling) - "BusyOut Trunk n TDM over IP failure (Active calls x Min y)". **Note:** Applicable only to Digital PSTN devices.
GWAPP_TRAP_BUSYOUT_PROXY_SET	Sent when the connection to the Proxy Set that is associated with this trunk/line is up/down ("BusyOut Trunk/Line n Proxy Set Failure").
GWAPP_TRAP_BUSYOUT_REGISTRATION	Sent when a failure occurs in server registration for this trunk/line ("BusyOut Trunk/Line n Registration Failure").

Trap	Description
GWAPP_TRAP_BUSYOUT_SERVING_IPGROUP	Sent when a failure occurs in a Serving IP Group for this trunk ("BusyOut Trunk n Serving IP Group Failure"). **Note:** Applicable only to Digital PSTN devices.
GWAPP_TRAP_PROXY_SET	Sent when a failure occurs in a Proxy Set (not per trunk/line, but per Proxy Set) - "Proxy Set ID n".
High Availability Traps **Note:** Applicable only to Mediant 3000 HA.	
acHASystemFaultAlarm	Sent when the High Availability (HA) system is faulty (i.e., no HA functionality).
acHASystemConfigMismatchAlarm	Sent when the configuration of the modules in the HA system is not identical, causing instability.
acHASystemSwitchOverAlarm	Sent when a switchover from the active to the redundant module has occurred.
acSWUpgradeAlarm	Sent for SW upgrade process errors.
PSTN - SONET Traps **Note:** Applicable only to Mediant 3000 with TP-6310.	
acSonetSectionLOFAlarm	SONET section Loss of Frame alarm.
acSonetSectionLOSAlarm	SONET section Loss of Signal alarm.
acSonetLineAISAlarm	SONET Line AIS alarm.
acSonetLineRDIAlarm	SONET Line RDI alarm.

In addition to the traps listed in the table above, the device also supports the following standard traps:

- **authenticationFailure**
- **coldStart**
- **linkDown**
- **linkup**
- **entConfigChange**
- **dsx1LineStatusChange** (Applicable only to Digital PSTN devices)
- **dsx3LineStatusChange** (Applicable only to Mediant 3000)

4.9 SNMP Interface Details

This subsection describes details of the SNMP interface needed when developing an Element Management System (EMS) for any of the TrunkPack-VoP Series products, or to manage a device with a MIB browser.

There are several alternatives for SNMP security:

- SNMPv2c community strings
- SNMPv3 User-based Security Model (USM) users
- SNMP encoded over IPSec (for more details, see "IPSec and IKE" on page 141)
- Various combinations of the above

Currently, both SNMP and *ini* file commands and downloads are not encrypted. For *ini* file encoding, refer to the device's *User's Manual*.

4.9.1 SNMP Community Names

By default, the device uses a single, read-only community string of "public" and a single read-write community string of "private". Up to five read-only community strings and up to five read-write community strings, and a single trap community string can be configured. Each community string must be associated with one of the following predefined groups:

Table 4-7: SNMP Predefined Groups

Group	Get Access	Set Access	Sends Traps
ReadGroup	Yes	No	Yes
ReadWriteGroup	Yes	Yes	Yes
TrapGroup	No	No	Yes

4.9.1.1 Configuring Community Strings via the Web

For detailed information on configuring community strings via the Web interface, refer to the device's *User's Manual*.

4.9.1.2 Configuring Community Strings via the ini File

The following *ini* file parameters are used to configure community strings:

- SNMPREADONLYCOMMUNITYSTRING_<x> = '#######'
- SNMPREADWRITECOMMUNITYSTRING_<x> = '#######'

Where <x> is a number from 0 through 4. Note that the '#' character represents any alphanumeric character. The maximum length of the string is 20 characters.

4.9.1.3 Configuring Community Strings via SNMP

To configure community strings, the EMS must use the standard snmpCommunityMIB. To configure the trap community string, the EMS must also use the snmpTargetMIB.

➢ To add a read-only v2user community string:

1. Add a new row to the snmpCommunityTable with CommunityName v2user.
2. Add a row to the vacmSecurityToGroupTable for SecurityName v2user, GroupName ReadGroup and SecurityModel snmpv2c.

➢ To delete the read-only v2user community string:

1. If v2user is being used as the trap community string, follow the procedure for changing the trap community string. (See below.)
2. Delete the snmpCommunityTable row with CommunityName v2user.
3. Delete the vacmSecurityToGroupTable row for SecurityName v2user, GroupName ReadGroup and SecurityModel snmpv2c.

➢ To add a read-write v2admin community string:

1. Add a new row to the snmpCommunityTable with CommunityName v2admin.
2. Add a row to the vacmSecurityToGroupTable for SecurityName v2admin, GroupName ReadWriteGroup and SecurityModel snmpv2c.

➢ To delete the read-write v2admin community string:

1. If v2admin is being used as the trap community string, follow the procedure for changing the trap community string. (See below.)
2. Delete the snmpCommunityTable row with a CommunityName of v2admin and GroupName of ReadWriteGroup.

➢ To change the only read-write community string from v2admin to v2mgr:

1. Follow the procedure above to add a read-write community string to a row for v2mgr.
2. Set up the EM such that subsequent set requests use the new community string, v2mgr.
3. If v2admin is being used as the trap community string, follow the procedure to change the trap community string. (See below.)
4. Follow the procedure above to delete a read-write community name in the row for v2admin.

The following procedure assumes that a row already exists in the snmpCommunityTable for the new trap community string. The trap community string can be part of the TrapGroup, ReadGroup, or ReadWriteGroup. If the trap community string is used solely for sending traps (recommended), then it should be made part of the TrapGroup.

➢ To change the trap community string:

1. Add a row to the vacmSecurityToGroupTable with these values: SecurityModel=2, SecurityName=the new trap community string, GroupName=TrapGroup, ReadGroup or ReadWriteGroup. The SecurityModel and SecurityName objects are row indices.

Note: You must add GroupName and RowStatus on the same set.

2. Modify the SecurityName field in the appropriate row of the snmpTargetParamsTable.
3. Remove the row from the vacmSecurityToGroupTable with SecurityName=the old trap community string.

4.9.2 SNMPv3 USM Users

You can configure up to 10 User-based Security Model (USM) users (referred to as *SNMPv3* user). Each SNMPv3 user can be configured for one of the following security levels:

Table 4-8: SNMPv3 Security Levels

Security Levels	Authentication	Privacy
noAuthNoPriv(1)	none	none
authNoPriv(2)	MD5 or SHA-1	none
authPriv(3)	MD5 or SHA-1	DES, 3DES, AES128, AES192, or AES256

Each SNMPv3 user must be associated with one of the predefined groups listed in the following table:

Table 4-9: SNMPv3 Predefined Groups

Group	Get Access	Set Access	Sends Traps	Security Level
ReadGroup1	Yes	No	Yes	noAuthNoPriv(1)
ReadWriteGroup1	Yes	Yes	Yes	noAuthNoPriv(1)
TrapGroup1	No	No	Yes	noAuthNoPriv(1)
ReadGroup2	Yes	No	Yes	authNoPriv(2)
ReadWriteGroup2	Yes	Yes	Yes	authNoPriv(2)
TrapGroup2	No	No	Yes	authNoPriv(2)
ReadGroup3	Yes	No	Yes	authPriv(3)
ReadWriteGroup3	Yes	Yes	Yes	authPriv(3)
TrapGroup3	No	No	Yes	authPriv(3)

4.9.2.1 Configuring SNMPv3 Users via the ini File

Use the SNMPUsers *ini* file table parameter to add, modify, and delete SNMPv3 users. The SNMPUsers *ini* table is a hidden parameter. Therefore, when you load the *ini* file to the device using the Web interface, the table is not included in the generated file.

Table 4-10: SNMPv3 Table Columns Description

Parameter	Description	Default
Row number	Table index. Its valid range is 0 to 9.	N/A
SNMPUsers_Username	Name of the v3 user. Must be unique. The maximum length is 32 characters.	N/A
SNMPUsers_AuthProtocol	Authentication protocol to be used for this user. Possible values are 0 (none), 1 (MD5), 2 (SHA-1)	0
SNMPUsers_PrivProtocol	Privacy protocol to be used for this user. Possible values are 0 (none), 1 (DES), 2 (3DES), 3 (AES128), 4 (AES192), 5 (AES256)	0
SNMPUsers_AuthKey	Authentication key.	""
SNMPUsers_PrivKey	Privacy key.	""
SNMPUsers_Group	The group that this user is associated with. Possible values are 0 (read-only group), 1 (read-write group), and 2 (trap group). The actual group will be ReadGroup<sl>, ReadWriteGroup<sl> or TrapGroup<sl> where <sl> is the SecurityLevel (1=noAuthNoPriv, 2=authNoPriv, 3=authPriv)	0

Keys can be entered in the form of a text password or in the form of a localized key in hex format. If using a text password, then it should be at least 8 characters in length. Below is an example showing the format of a localized key:

```
26:60:d8:7d:0d:4a:d6:8c:02:73:dd:22:96:a2:69:df
```

The following sample configuration creates three SNMPv3 USM users.

```
[ SNMPUsers ]
FORMAT SNMPUsers_Index = SNMPUsers_Username,
SNMPUsers_AuthProtocol, SNMPUsers_PrivProtocol, SNMPUsers_AuthKey,
SNMPUsers_PrivKey, SNMPUsers_Group;
SNMPUsers 0 = v3user, 0, 0, -, -, 0;
SNMPUsers 1 = v3admin1, 1, 0, myauthkey, -, 1;
SNMPUsers 2 = v3admin2, 2, 1, myauthkey, myprivkey, 1;
[ \SNMPUsers ]
```

The example above creates three SNMPv3 users:

- The user v3user is set up for a security level of noAuthNoPriv(1) and is associated with ReadGroup1.
- The user v3admin1 is setup for a security level of authNoPriv(2), with authentication protocol MD5. The authentication text password is “myauthkey” and the user is associated with ReadWriteGroup2.
- The user v3admin2 is setup for a security level of authPriv(3), with authentication protocol SHA-1 and privacy protocol DES. The authentication text password is “myauthkey”, the privacy text password is “myprivkey”, and the user is associated with ReadWriteGroup3.

4.9.2.2 Configuring SNMPv3 Users via SNMP

To configure SNMPv3 users, the EMS must use the standard snmpUsmMIB and the snmpVacmMIB.

➢ **To add a read-only, noAuthNoPriv SNMPv3 user, v3user:**

1. Clone the row with the same security level. After the clone step, the status of the row will be notReady(3).
2. Activate the row. That is, set the row status to active(1).
3. Add a row to the vacmSecurityToGroupTable for SecurityName v3user, GroupName ReadGroup1 and SecurityModel usm(3).

Note: A row with the same security level (noAuthNoPriv) must already exist in the usmUserTable. (see the usmUserTable for details).

➢ **To delete the read-only, noAuthNoPriv SNMPv3 user, v3user:**

1. If v3user is associated with a trap destination, follow the procedure for associating a different user to that trap destination. (See below.)
2. Delete the vacmSecurityToGroupTable row for SecurityName v3user, GroupName ReadGroup1 and SecurityModel usm.
3. Delete the row in the usmUserTable for v3user.

➢ **To add a read-write, authPriv SNMPv3 user, v3admin1:**

1. Clone the row with the same security level.
2. Change the authentication key and privacy key.
3. Activate the row. That is, set the row status to active(1).
4. Add a row to the vacmSecurityToGroupTable for SecurityName v3admin1, GroupName ReadWriteGroup3 and SecurityModel usm(3).

Note: A row with the same security level (authPriv) must already exist in the usmUserTable (see the usmUserTable for details).

➢ **To delete the read-write, authPriv SNMPv3 user, v3admin1:**

1. If v3admin1 is associated with a trap destination, follow the procedure for associating a different user to that trap destination. (See below.)
2. Delete the vacmSecurityToGroupTable row for SecurityName v3admin1, GroupName ReadWriteGroup1 and SecurityModel usm.
3. Delete the row in the usmUserTable for v3admin1.

4.9.3 Trusted Managers

By default, the SNMP agent accepts Get and Set requests from any IP address, as long as the correct community string is used in the request. Security can be enhanced implementing *Trusted Managers*. A Trusted Manager is an IP address from which the SNMP agent accepts and processes Get and Set requests. An element management can be used to configure up to five Trusted Manager.

The concept of Trusted Managers is considered to be a weak form of security and therefore is not a required part of SNMPv3 security, which uses authentication and privacy. Trusted Managers for the devices' SNMP agent are applicable only for SNMPv2c users. An exception to this is when the community string is not the default string (‘public’/’private’), at which time Trusted Managers are applicable for SNMPV2c users alongside SNMPv3 users.

> **Note:** If trusted managers are defined, then all community strings works from all trusted managers, i.e.,there is no way to associate a community string with specific trusted managers.

4.9.3.1 Configuring Trusted Managers via ini File

To set the Trusted Managers table from start up, write the following in the *ini* file:

```
SNMPTRUSTEDMGR_X = D.D.D.D
```

Where *X* is any integer between 0 and 4 (0 sets the first table entry, 1 sets the second and so on), and *D* is an integer between 0 and 255.

4.9.3.2 Configuring Trusted Managers via SNMP

To configure Trusted Managers, the EMS must use the SNMP-COMMUNITY-MIB and snmpCommunityMIB and the snmpTargetMIB.

The procedure below assumes the following: at least one configured read-write community; currently no Trusted Managers; TransportTag for columns for all snmpCommunityTable rows are currently empty.

➢ **To add the first Trusted Manager:**

1. Add a row to the snmpTargetAddrTable with these values: Name=mgr0, TagList=MGR, Params=v2cparams.
2. Add a row to the snmpTargetAddrExtTable table with these values: Name=mgr0, snmpTargetAddrTMask=255.255.255.255:0. The agent does not allow creation of a row in this table unless a corresponding row exists in the snmpTargetAddrTable.
3. Set the value of the TransportTag field on each non-TrapGroup row in the snmpCommunityTable to MGR.

The procedure below assumes the following: at least one configured read-write community; currently one or more Trusted Managers; TransportTag for columns for all rows in the snmpCommunityTable are currently set to MGR. This procedure must be done from one of the existing Trusted Managers.

➢ To add a subsequent Trusted Manager:

1. Add a row to the snmpTargetAddrTable with these values: Name=mgrN, TagList=MGR, Params=v2cparams, where N is an unused number between 0 and 4.
2. Add a row to the snmpTargetAddrExtTable table with these values: Name=mgrN, snmpTargetAddrTMask=255.255.255.255:0.

An alternative to the above procedure is to set the snmpTargetAddrTMask column while you are creating other rows in the table.

The procedure below assumes the following: at least one configured read-write community; currently two or more Trusted Managers; taglist for columns for all rows in the snmpCommunityTable are currently set to MGR. This procedure must be done from one of the existing trusted managers, but not the one that is being deleted.

➢ To delete a Trusted Manager (not the last one):

- Remove the appropriate row from the snmpTargetAddrTable.

The change takes affect immediately. The deleted trusted manager cannot access the device. The agent automatically removes the row in the snmpTargetAddrExtTable.

The procedure below assumes the following: at least one configured read-write community; currently only one Trusted Manager; taglist for columns for all rows in the snmpCommunityTable are currently set to MGR. This procedure must be done from the final Trusted Manager.

➢ To delete the last Trusted Manager:

1. Set the value of the TransportTag field on each row in the snmpCommunityTable to the empty string.
2. Remove the appropriate row from the snmpTargetAddrTable.

The change takes affect immediately. All managers can now access the device. The agent automatically removes the row in the snmpTargetAddrExtTable.

4.9.4 SNMP Ports

The SNMP Request Port is 161 and Trap Port is 162. These port numbers for SNMP requests and responses can be changed by using the following *ini* file parameter:

SNMPPort = <port_number>

The valid value is any valid UDP port number; the default is 161 (recommended).

4.9.5 Multiple SNMP Trap Destinations

An agent can send traps to up to five managers. For each manager you need to define the manager IP address and trap receiving port along with enabling the sending to that manager. You can also associate a trap destination with a specific SNMPv3 USM user. Traps are sent to this trap destination using the SNMPv3 format and the authentication and privacy protocol configured for that user.

To configure the Trap Managers table, use one of the following methods:

- Web interface (refer to the device's *User's Manual*)
- *ini* file (see "Configuring Trap Managers via the ini File" on page 77)
- SNMP (see "Configuring Trap Managers via SNMP" on page 78)

4.9.5.1 Configuring Trap Managers via Host Name

One of the five available SNMP managers can be defined using the manager's host name (i.e., FQDN). This is currently supported using an *ini* file only (SNMPTrapManagerHostName).

When this parameter value is defined for this trap, the device at start up tries to resolve the host name. Once the name is resolved (i.e., the IP address is found), the resolved IP address replaces the last entry of the trap manager table (defined by the parameter SNMPManagerTableIP_x) and the last trap manager entry of snmpTargetAddrTable in the snmpTargetMIB. The port is 162 (unless specified otherwise). The row is marked as 'used' and the sending is 'enabled'.

When using 'host name' resolution, any changes made by the user to this row in either MIBs are overwritten by the device when a resolving is redone (once an hour).

Note: Some traps may be lost until the name resolving is complete.

4.9.5.2 Configuring Trap Managers via ini File

In the *ini* file, parameters below can be set to enable or disable the sending of SNMP traps. Multiple trap destinations can be supported on the device by setting multiple trap destinations in the ini file.

- **SNMPManagerTrapSendingEnable_<x>:** indicates whether or not traps are to be sent to the specified SNMP trap manager. A value of ‘1’ means that it is enabled, while a value of ‘0’ means disabled. The <x> represents a number 0, 1, or 2, which is the array element index. Currently, up to five SNMP trap managers is supported.

- **SNMPManagerTrapUser_<x>:** indicates to send an SNMPv2 trap using the trap user community string configured with the SNMPTrapCommunityString parameter. You may instead specify an SNMPv3 user name.

Below is an example of entries in the *ini* file regarding SNMP. The device can be configured to send to multiple trap destinations.

```
; SNMP trap destinations
; The device maintains a table of trap destinations containing 5
; rows. The rows are numbered 0..4. Each block of 5 items below
; applies to a row in the table.
;
; To configure one of the rows, uncomment all 5 lines in that
; block. Supply an IP address and if necessary, change the port
; number.
;
; To delete a trap destination, set ISUSED to 0.
;
;SNMPManagerTableIP_0=
;SNMPManagerTrapPort_0=162
;SNMPManagerIsUsed_0=1
;SNMPManagerTrapSendingEnable_0=1
;SNMPManagerTrapUser_0=’’
;
;SNMPManagerTableIP_1=
;SNMPManagerTrapPort_1=162
;SNMPManagerIsUsed_1=1
;SNMPManagerTrapSendingEnable_1=1
;SNMPMANAGERTRAPUSER_1=’’
;
;SNMPManagerTableIP_2=
;SNMPManagerTrapPort_2=162
;SNMPManagerIsUsed_2=1
;SNMPManagerTrapSendingEnable_2=1
;SNMPManagerTrapUser_2=’’
;
;SNMPManagerTableIP_3=
;SNMPManagerTrapPort_3=162
;SNMPManagerIsUsed_3=1
;SNMPManagerTrapSendingEnable_3=1
;SNMPManagerTrapUser_3=’’
;
;SNMPMANAGERTABLEIP_4=
;SNMPManagerTrapPort_4=162
;SNMPManagerIsUsed_4=1
;SNMPManagerTrapSendingEnable_4=1
;SNMPManagerTrapUser_4=’’
```

The ‘trap manager host name’ is configured via SNMPTrapManagerHostName. For example:

```
;SNMPTrapManagerHostName = 'myMananger.corp.MyCompany.com'
```

Note: The same information that is configurable in the *ini* file can also be configured via the acBoardMIB.

4.9.5.3 Configuring SNMP Engine ID

The SNMPEngineIDString *ini* file parameter configures the SNMP engine ID. The ID can be a string of up to 36 characters. Once defined, the device must be reset for the parameter to take effect.

The default value is 00:00:00:00:00:00:00:00:00:00:00:00 (12 Hex characters). The provided key must be set with 12 Hex values delimited by ‘:’.

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 then generated, according to RFC 3411.

Before setting this parameter, all SNMPv3 users must be deleted, otherwise the configuration is ignored.

4.9.5.4 Configuring Trap Managers via SNMP

The snmpTargetMIB interface is available for configuring trap managers.

> **Note:** The acBoard MIB is planned to become obsolete. The only relevant section in this MIB is the trap subtree acTRap.

➢ To add an SNMPv2 trap destination:

- Add a row to the snmpTargetAddrTable with these values: Name=trapN, TagList=AC_TRAP, Params=v2cparams, where N is an unused number between 0 and 4

All changes to the trap destination configuration take effect immediately.

➢ To add an SNMPv3 trap destination:

1. Add a row to the snmpTargetAddrTable with these values: Name=trapN, TagList=AC_TRAP, Params=usm<user>, where N is an unused number between 0 and 4, and <user> is the name of the SNMPv3 that this user is associated with.
2. If a row does not already exist for this combination of user and SecurityLevel, add a row to the snmpTargetParamsTable with these values: Name=usm<user>, MPModel=3(SNMPv3), SecurityModel=3 (usm), SecurityName=<user>, SecurityLevel=M, where M is either 1(noAuthNoPriv), 2(authNoPriv) or 3(authPriv).

All changes to the trap destination configuration take effect immediately.

➢ To delete a trap destination:

- Remove the appropriate row from the snmpTargetAddrTable.
- If this is the last trap destination associated with this user and security level, you could also delete the appropriate row from the snmpTargetParamsTable.

➢ To modify a trap destination:

You can change the IP address and or port number for an existing trap destination. The same effect can be achieved by removing a row and adding a new row.

- Modify the IP address and/or port number for the appropriate row in the snmpTargetAddrTable.

➢ To disable a trap destination:

- Change TagList on the appropriate row in the snmpTargetAddrTable to the empty string.

➢ To enable a trap destination:

- Change TagList on the appropriate row in the snmpTargetAddrTable to ‘AC_TRAP’.
- Change TagList on the appropriate row in the snmpTargetAddrTable to "AC_TRAP".

4.10 Dual Module Interface

Note: This subsection is applicable only to 2000 Series devices.

Dual module blades have a first and second module (the first is on the right side of the blade -- TP-1610 and IPM-1610 -- when looking at it from the front). Differentiation is based on the modules' serial numbers.

MIB object acSysIdSerialNumber always returns the serial number of the module on which the GET is performed. MIB object acSysIdFirstSerialNumber always returns the serial number of the first module.

If the module on which the GET is performed is the second module, the values in these two are different. If, on the other hand, the module is the first module, the value in the two objects is the same.

4.11 SNMP NAT Traversal

A NAT placed between the device and the element manager calls for traversal solutions:

- **Trap source port:** all traps are sent from the SNMP port (default is 161). A manager receiving these traps can use the binding information (in the UDP layer) to traverse the NAT back to the device.
  The trap destination address (port and IP) are as configured in the snmpTargetMIB.

- **acKeepAliveTrap:** this trap is designed to be a constant life signal from the device to the manager, allowing the manager NAT traversal at all times. The acBoardTrapGlobalsAdditionalInfo1 varbind has the device’s serial number.

  The destination port (i.e., the manager port for this trap), can be set to be different than the port to which all other traps are sent. To do this, use the **acSysSNMPKeepAliveTrapPort** object in the acSystem MIB or the KeepAliveTrapPort *ini* file parameter.

  The Trap is instigated in three ways:

  - Via an *ini* file parameter (SendKeepAliveTrap = 1). This ensures that the trap is continuously sent. The frequency is set via the 9/10 of the NATBindingDefaultTimeout (or MIB object acSysSTUNBindingLifeTime) parameter.
  - After the STUN client has discovered a NAT (any NAT).
  - If the STUN client can not contact a STUN server.

Note: The two latter options require the STUN client be enabled (*ini* file parameter EnableSTUN). In addition, once the acKeepAlive trap is instigated it does not stop.

- The manager can view the NAT type in the MIB:
  audioCodes(5003).acProducts(9).acBoardMibs(10).acSystem(10).acSystemStatus(2).
  acSysNetwork(6).acSysNAT(2).acSysNATType(1)

- The manager also has access to the STUN client configuration: audioCodes(5003).acProducts(9).acBoardMibs(10).acSystem(10).acSystemConfiguration(1).acSysNetworkConfig(3).acSysNATTraversal(6).acSysSTUN(21)
- **acNATTraversalAlarm**: When the NAT is placed in front of a device that is identified as a symmetric NAT, this alarm is raised. It is cleared when a non-symmetric NAT or no NAT replaces the symmetric one.

4.12 Active Alarm Severity Levels

The acSysStateGWSeverity parameter reflects the highest active alarm severity on the device. The options include the following:

- noAlarm(0)
- indeterminate(1)
- warning(2)
- minor(3)
- major(4)
- critical(5)

4.13 Media Server Configuration

Note: This subsection is applicable only to IPmedia Series and Mediant 1000.

Configuration for the device can be performed by using the SNMP interfaces in the acBoardMIB or setting of configuration parameters in the *ini* file. Access to the configuration parameters is also provided through the Web interface.

A default *ini* (or initialization) template has been defined, which configures the configuration parameters to settings that typically, do not require later modificatons.

Configuration parameters in the acBoardMIB specific to services on the device include:

- **amsApsIpAddress:** IP address of the audio provisioning server
- **amsApsPort**: port number to use for the audio provisioning server
- **amsPrimaryLanguage:** primary language used for audio variables
- **amsSecondaryLanguage:** secondary language used for audio variables

4.14 Systems

Note: This subsection is applicable only to 3000 Series.

For the management of a system (a chassis with more then one type of module running), the acSystem/acSystemChassis subtree in the acSystem MIB should be used:

- The first few objects are scalars that are read-only objects for the dry-contacts' state.
- **acSysModuleTable:** A table containing mostly status information that describes the blade modules in the system. In addition, the table can be used to reset an entire system, reset a redundant module or perform switchover when the system is HA.
- **acSysFanTrayTable**: A status-only table with the fan tray's state. Objects in the table indicate the specific state of the individual fans within the fan tray.
- **acSysPowerSupplyTable**: A status-only table with the states of the two power supplies.
- **acSysPEMTable**: A status-only table with the states of the two PEMs (Power Entry Modules).

The above tables are complemented by the following alarm traps (as defined in the acBoard MIB. For more details, see "SNMP Traps" on page 84):

- **acFanTrayAlarm**: fault in the fan tray or fan tray missing.
- **acPowerSupplyAlarm**: fault in one of the power supply modules or PS module missing.
- **acPEMAlarm**: fault in the one of the PEM modules or PEM module missing.
- **acSAMissingAlarm**: SA module missing or non operational.
- **acUserInputAlarm**: the alarm is raised when the input dry contact is short circuited and cleared when the circuit is reopened.

4.15 High Availability Systems

Note: This subsection is applicable only to Mediant 3000.

For the management of the High Availability (HA) systems, use the acSysChassis MIB subtree (as in the above section). The acSysModuleTable gives the HA state of the system. This includes defining which modules are active and which are in standby mode (redundant). The table also enables to read some of the statuses of the redundant modules (such as SW version, HW version, temperature, license key list, etc.). Resetting the system, resetting the redundant module, and performing switchover are performed done using this table.

Complementing the above are the following alarm traps (as defined in the acBoard MIB):

- **acHASystemFaultAlarm:** the HA is faulty and therefore, there is no HA.

- **acHASystemConfigMismatchAlarm**: configuration to the modules in the HA system us uneven causing instability.
- **acHASystemSwitchOverAlarm**: a switchover from the active to the redundant module has occurred.

4.16 Configuring Clock Synchronization

> **Note:** This subsection is applicable only to Mediant 3000.

The procedures below describe how to configure clock synchronization modes.

➢ To configure line synchronization, perform the following steps:

1. Set acSysTimingMode to lineSync.
2. Set acSysTDMClockSource to the interface (according to the hardware you are using) from which you wish to derive the clock.
3. Set TDMBusLocalReference to the reference trunk number.
4. Set acSysTDMClockPLLOutOfRange to the requested value.
5. Set acSysActionSetOnLineChangesApply to 1 in order to apply all changes.

➢ To configure BITS Synchronization mode through SNMP:

1. Set acSysTimingMode to external.
2. Set acSysTDMClockBitsReference (1 – Primary Clock Reference is BITs A. (Default) 2 – Primary Clock Reference is BITs B).
3. Set acSysTDMClockEnableFallBack (manual(0), autoNon-Revertive(1), auto-Revertive(2) TDMBusEnableFallback sets the fallback clock method between primary to secondary BITS clock references.)
4. Set acSysTimingExternalIFType to define the external BITS reference transmission type for both primary and secondary interfaces.
5. Set acSysTimingT1LineBuildOut / acSysTimingE1LineBuildOut.
6. Set acSysTimingValidationTime to the requested time range: 0-15 minutes.
7. Set acSysActionSetOnLineChangesApply to 1 in order to apply all changes.

4.17 SNMP Administrative State Control

Node maintenance for the device is provided via an SNMP interface. The acBoardMIB provides two parameters for graceful and forced shutdowns of the device. These parameters are in the acBoardMIB as follows:

- acSysActionAdminState - read-write MIB object. When a GET request is sent for this object, the agent returns the current device administrative state - determines the device's desired operational state:

    - **locked (0):** Shutdown the device in the time frame set by acSysActionAdminStateLockTimeout.
    - **shuttingDown (1):** (read-only) Graceful shutdown is being performed - existing calls are allowed to complete, but no new calls are allowed.
    - **unlocked (2):** The device is in service.

    On a SET request, the manager supplies the required administrative state, either locked(0) or unlocked(2). When the device changes to either shuttingDown or locked state, an adminStateChange alarm is raised. When the device changes to an unlocked state, the adminStateChange alarm is cleared.

- acSysActionAdminStateLockTimeout - defines the time remaining (in seconds) for the shutdown to complete:
    - **0:** immediate shutdown and calls are terminated (forced lock)
    - **1:** waits until all calls are terminated (i.e., perform a Graceful shutdown)
    - **> 0:** the number of seconds to wait before the graceful shutdown turns into a force lock

Note: The acSysActionAdminStateLockTimeout must be set before the acSysActionAdminState.

4.18 AudioCodes' Element Management System

Using AudioCodes' Element Management System (EMS) is recommended for customers requiring large deployments (for example, multiple devices in globally distributed enterprise offices) that need to be managed by central personnel.

The EMS is not included in the device's supplied package. Contact AudioCodes for detailed information on AudioCodes' EMS solution for large VoIP deployments.

4.19 SNMP Traps

This subsection provides information on proprietary SNMP traps supported by the device. There is a separation between traps that are alarms and traps that are not (i.e., logs). All the traps have the same structure made up of the same 11 varbinds (Variable Binding), i.e., 1.3.6.1.4.1.5003.9.10.1.21.1. For a list of the varbinds, see "Trap Varbinds" on page 121.

The source varbind is composed of a string that details the device component from which the trap is being sent (forwarded by the hierarchy in which it resides). For example, an alarm from an SS7 link has the following string in its source varbind:

acBoard#1/SS7#0/SS7Link#6

In this example, the SS7 link number is specified as 6 and is part of the only SS7 module in the device that is placed in slot number 1 (in a chassis) and is the module to which this trap relates. For devices where there are no chassis options the slot number of the device is always 1.

4.19.1 SNMP Alarms in Syslog

All SNMP alarms are sent to the Syslog server using the following format.

- **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 4-11: 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 alarm:**

  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:

4.19.2 Alarm Traps

The tables in the following subsections provide information on alarms that are raised as a result of a generated SNMP trap. The component name (described in each of the following headings) refers to the string that is provided in the acBoardTrapGlobalsSource trap varbind. To clear a generated alarm, the same notification type is sent but with the severity set to 'cleared'.

4.19.2.1 Chassis SNMP Alarms

Note: These alarms are applicable only to 3000 Series, Mediant 1000, and Mediant 600.

The source varbind text for the alarm under this component is Chassis#0/FanTray#0.

Table 4-12: acFanTrayAlarm (Applicable Only to 3000 Series and Mediant 1000)

Alarm:	acFanTrayAlarm
OID:	1.3.6.1.4.1.5003.9.10.1.21.2.0.29
Default Severity:	Critical
Event Type:	equipmentAlarm
Probable Cause:	heatingVentCoolingSystemProblem
Alarm Text:	Fan-Tray Alarm
Status Changes:	
1. Condition:	Fan-Tray is missing
Alarm Status:	Critical
<text> Value:	Fan-Tray Alarm. Fan-Tray is missing
2. Condition:	One or more fans in the Fan-Tray are faulty.
Alarm Status:	Major
Corrective Action:	Fan is faulty
3. Condition:	Fan tray is in place and fans are working.
Alarm Status:	Cleared

The source varbind text for the alarm under this component is Chassis#0/PowerSupply#<m>, where *m* is the power supply's slot number.

Table 4-13: acPowerSupplyAlarm (Applicable Only to 3000 Series and Mediant 1000)

Alarm:	acPowerSupplyAlarm
OID:	1.3.6.1.4.1.5003.9.10.1.21.2.0.30
Default Severity:	Critical
Event Type:	equipmentAlarm
Probable Cause:	powerProblem
Alarm Text:	Power-Supply Alarm. Power-Supply is missing.
Status Changes:	
1. Condition:	The HA (High Availability) feature is active (applicable only to Mediant 3000) and one of the power supply units is faulty or missing.
Alarm Status:	Major
2. Condition:	PS unit is placed and working.
Alarm Status:	Cleared

The source varbind text for the alarm under this component is Chassis#0.

Table 4-14: acUserInputAlarm

Alarm:	acUserInputAlarm
OID:	1.3.6.1.4.1.5003.9.10.1.21.2.0.36
Default Severity:	Critical
Event Type:	equipmentAlarm
Probable Cause:	inputDeviceError
Alarm Text:	User input Alarm. User's Input-Alarm turn on.
Status Changes:	
1. Condition:	Input dry contact is short circuited.
Alarm Status:	Critical
2. Condition:	Input dry contact circuit is reopened.
Alarm Status:	Cleared

The following trap is applicable only to the 3000 Series devices. The source varbind text for the alarm under this component is Chassis#0/PemCard#<m>, where *m* is the power entry module's (PEM) slot number.

Table 4-15: acPEMAlarm (Applicable Only to 3000 Series)

Alarm:	acPEMAlarm
OID:	1.3.6.1.4.1.5003.9.10.1.21.2.0.31
Default Severity:	Critical
Event Type:	equipmentAlarm
Probable Cause:	underlyingResourceUnavailable
Alarm Text:	PEM Module Alarm.
Status Changes:	
1. Condition:	The HA (High Availability) feature is active (applicable only to Mediant 3000) and one of the PEM units is missing (PEM – Power Entry Module)
Alarm status:	Critical
<text> Value:	PEM card is missing.
2. Condition:	PEM card is placed and both DC wires are in.
Alarm Status:	Cleared

The following trap is applicable only to Mediant 1000. The source varbind text for the alarm under this component is Chassis#0/module#<m>, where *m* is the module's number.

Table 4-16: acHwFailureAlarm (Applicable Only to Mediant 1000 and Mediant 600)

Alarm:	acHwFailureAlarm
OID:	1.3.6.1.4.1.5003.9.10.1.21.2.0.43
Default Severity:	Critical
Event Type:	equipmentAlarm
Probable Cause:	equipmentMalfunction
Alarm Text:	Module Alarm: <text>
Status Changes:	
1. Condition:	The module is faulty or has been removed incorrectly.
Alarm Status:	Critical
<text> Value:	Faulty IF-Module
Note:	This alarm is not cleared. The device must be restarted to clear this alarm.
2. Condition:	Module mismatch - module and CPU board mismatch.
Alarm Status:	Major
<text> Value:	IF-Module Mismatch
Note:	This alarm is not cleared. The device must be restarted to clear this alarm.

4.19.2.2 Timing Module Alarms

Note: These alarms are applicable only to Mediant 3000.

The Timing Module traps below use the following source varbind text: Chassis#0/TimingManager#0

Table 4-17: acTMInconsistentRemoteAndLocalPLLStatus Alarm

Alarm:	acTMInconsistentRemoteAndLocalPLLStatus
OID:	1.3.6.1.4.1.5003.9.10.1.21.2.0.56
Default Severity:	Major
Event Type:	equipmentAlarm
Probable Cause:	underlyingResourceUnavailable
Alarm Text:	Timing Manager Alarm <text>
1. Condition:	The alarm is triggered when the system is in 1+1 status and redundant board PLL status is deferent than active board PLL status
Alarm Status:	Major
<text> Value:	Timing Manager Alarm. Local and Remote PLLs status is different.
2. Condition:	
Alarm Status:	Status remains major until a reboot. A clear trap is not sent.
Corrective Action:	Synchronize the timing module.

Table 4-18: acTMReferenceStatus Alarm

Alarm:	acTMReferenceStatus
OID:	1.3.6.1.4.1.5003.9.10.1.21.2.0.57
Default Severity:	Major
Event Type:	equipmentAlarm
Probable Cause:	underlyingResourceUnavailable
Alarm Text:	Timing Manager Alarm <text>
Status Changes:	While primary and secondary clock references are down for more than 24 hours, the alarm will be escalated to critical.
1. Condition:	The alarm is triggered when the primary reference or secondary reference or both are down.
Alarm Status:	Major
<text> Value:	Timing Manager Alarm. PRIMARY REFERENCE DOWN/SECONDARY REFERENCE DOWN/ALL REFERENCES ARE DOWN
2. Condition:	
Alarm Status:	Status remains major until a reboot. A clear trap is not sent.
Corrective Action:	Synchronize the timing module.

Table 4-19: acTMReferenceChange Alarm

Alarm:	acTMReferenceChange
OID:	1.3.6.1.4.1.5003.9.10.1.21.2.0.58
Default Severity:	Indeterminate
Event Type:	
Probable Cause:	
Alarm Text:	Timing Manager
Status Changes:	
1. Condition:	Log is sent on PLL status change.

4.19.2.3 Trunk Alarms

Note: These alarms are applicable only to Digital PSTN devices.

The source varbind text for the alarms under the component below is Interfaces#0/Trunk#<m>, where *m* is the trunk IF number, 1 being the first trunk.

Table 4-20: acTrunksAlarmNearEndLOS

Alarm:	acTrunksAlarmNearEndLOS
OID:	1.3.6.1.4.1.5003.9.10.1.21.2.0.49
Default Severity:	Critical
Event Type:	communicationsAlarm
Probable Cause:	lossOfSignal
Alarm Text:	Trunk LOS Alarm.
Status Changes:	
Condition:	Near-end LOS
Alarm Status:	Critical
Condition:	End of LOS
Alarm Status:	Cleared
Corrective Action:	Ensure the trunk is properly connected.

Table 4-21: acTrunksAlarmNearEndLOF

Alarm:	acTrunksAlarmNearEndLOF
OID:	1.3.6.1.4.1.5003.9.10.1.21.2.0.50
Default Severity:	Critical
Event Type:	communicationsAlarm
Probable Cause:	lossOfFrame
Alarm Text:	Trunk LOF Alarm.
Status Changes:	
Condition:	Near end LOF
Alarm Status:	Critical
Condition:	End of LOF
Alarm Status:	Cleared
Corrective Action:	Ensure the trunk is connected to a proper follow up device. Ensure correct clocking setup.

Table 4-22: acTrunksAlarmRcvAIS

Alarm:	acTrunksAlarmRcvAIS
OID:	1.3.6.1.4.1.5003.9.10.1.21.2.0.51
Default Severity:	Critical
Event Type:	communicationsAlarm
Probable Cause:	receiveFailure
Alarm Text:	Trunk AIS Alarm
Status Changes:	
Condition:	Receive AIS
Alarm Status:	Critical
Condition:	End of AIS
Alarm Status:	Cleared
Corrective Action:	None

Table 4-23: acTrunksAlarmFarEndLOF

Alarm:	acTrunksAlarmFarEndLOF
OID:	1.3.6.1.4.1.5003.9.10.1.21.2.0.52
Default Severity:	Critical
Event Type:	communicationsAlarm
Probable Cause:	transmitFailure
Alarm Text:	Trunk RAI Alarm.
Status Changes:	
Condition:	RAI
Alarm Status:	Critical
Condition:	End of RAI
Alarm Status:	Cleared
Corrective Action:	Ensure correct transmission.

Table 4-24: dsx1LineStatusChange

Alarm:	dsx1LineStatusChange
OID:	1.3.6.1.2.1.10.18.15.0.1
Default Severity:	Major on raise; Clear on clear
Event Type:	communicationsAlarm
Probable Cause:	
Alarm Text:	DS1 Line Status
Status Changes:	
Additional Info1,2,3:	Updated DS1 Line Status. This variable indicates the Line Status of the interface. It contains loopback, failure, received 'alarm' and transmitted 'alarms' information. The dsx1LineStatus is a bit map represented as a sum, therefore, it can represent multiple failures (alarms) and a LoopbackState simultaneously. dsx1NoAlarm must be set if and only if no other flag is set. If the dsx1loopbackState bit is set, the loopback in effect can be determined from the dsx1loopbackConfig object. The various bit positions are: 1 dsx1NoAlarm No alarm present 2 dsx1RcvFarEndLOF Far end LOF (a.k.a., Yellow Alarm) 4 dsx1XmtFarEndLOF Near end sending LOF Indication 8 dsx1RcvAIS Far end sending AIS 16 dsx1XmtAIS Near end sending AIS 32 dsx1LossOfFrame Near end LOF (a.k.a., Red Alarm) 64 dsx1LossOfSignal Near end Loss Of Signal 128 dsx1LoopbackState Near end is looped 256 dsx1T16AIS E1 TS16 AIS 512 dsx1RcvFarEndLOMF Far End Sending TS16 LOMF 1024 dsx1XmtFarEndLOMF Near End Sending TS16 LOMF 2048 dsx1RcvTestCode Near End detects a test code 4096 dsx1OtherFailure Any line status not defined here 8192 dsx1UnavailSigState Near End in Unavailable Signal State 16384 dsx1NetEquipOOS Carrier Equipment Out of Service 32768 dsx1RcvPayloadAIS DS2 Payload AIS 65536 dsx1Ds2PerfThreshold DS2 Performance Threshold Exceeded

4.19.2.4 SONET Alarms

Note: These alarms are applicable only to Mediant 3000 with TP-6310 blade.

The source varbind text for the alarms under this component is Interfaces#0/Sonet#<m>, where *m* is the Sonet IF number.

Table 4-25: AcSonetSectionLOFAlarm

Alarm:	acSonetSectionLOFAlarm
OID:	1.3.6.1.4.1.5003.9.10.1.21.2.0.38
Default Severity:	Critical
Event Type:	communicationsAlarm
Probable Cause:	lossOfFrame
Alarm Text:	SONET-Section LOF.
Status Changes:	
1. Condition:	LOF condition is present on SONET no.n
Alarm Status:	Critical
<text> Value:	LOF
Note:	The sonetSectionCurrentStatus field in the sonetSectionCurrentTable will have a value sonetSectionLOF (4).
2. Condition:	LOF condition is not present.
Alarm Status:	Cleared

Table 4-26: AcSonetSectionLOSAlarm

Alarm:	acSonetSectionLOSAlarm
OID:	1.3.6.1.4.1.5003.9.10.1.21.2.0.39
Default Severity:	critical
Event Type:	communicationsAlarm
Probable Cause:	lossOfSignal
Alarm Text:	SONET-Section LOS.
Status Changes:	
1. Condition:	LOS condition is present on SONET no #n
Alarm Status:	Critical
<text> Value:	LOS
Note:	The sonetSectionCurrentStatus field in the sonetSectionCurrentTable will have a value sonetSectionLOS (2).
2. Condition:	AIS condition is present (LOS condition is not present)
Alarm Status:	Critical
3. Condition:	LOS condition is not present.
Alarm Status:	Cleared

Table 4-27: AcSonetLineAISAlarm

Alarm:	acSonetLineAISAlarm
OID:	1.3.6.1.4.1.5003.9.10.1.21.2.0.40
Default Severity:	Critical
Event Type:	communicationsAlarm
Probable Cause:	receiveFailure
Alarm Text:	SONET-Line AIS.
Status Changes:	
1. Condition:	AIS condition is present on SONET-Line #n.
Alarm Status:	Critical
<text> Value:	AIS
Note:	The sonetLineCurrentStatus field in the sonetLineCurrentTable will have a value sonetLineAIS (2).
2. Condition:	AIS condition is not present.
Alarm Status:	Cleared

Table 4-28: AcSonetLineRDIAlarm

Alarm:	acSonetLineRDIAlarm
OID:	1.3.6.1.4.1.5003.9.10.1.21.2.0.41
Default Severity:	Critical
Event Type:	communicationsAlarm
Probable Cause:	transmitFailure
Alarm Text:	SONET-Line RDI.
Status Changes:	
1. Condition:	RDI condition is present on SONET-Line #n.
Alarm Status:	Critical
<text> Value:	RDI
Note:	The sonetLineCurrentStatus field in the sonetLineCurrentTable will have a value sonetLineRDI (4).
2. Condition:	RDI condition is not present.
Alarm Status:	Cleared

The source varbind text for the alarms under this component is Interfaces#0/Path#<m>, where *m* is the Sonet IF number.

Table 4-29: acSonetPathSTSLOPAlarm

Alarm:	acSonetPathSTSLOPAlarm
OID:	1.3.6.1.4.1.5003.9.10.1.21.2.0.61
Default Severity:	Critical
Event Type:	communicationsAlarm
Probable Cause:	receiveFailure
Alarm Text:	Sonet Path STS AIS alarm.
Status Changes:	
1. Condition:	LOP condition is present on Path #n.
Alarm Status:	Critical
<text> Value:	LOP
Note:	The sonetPathCurrentStatus in the sonetPathCurrentTable has a value of sonetPathSTSLOP (2).
2. Condition:	LOP condition is not present.
Alarm Status:	Cleared

Table 4-30: acSonetPathSTSAISAlarm

Alarm:	acSonetPathSTSAISAlarm
OID:	1.3.6.1.4.1.5003.9.10.1.21.2.0.62
Default Severity:	Critical
Event Type:	communicationsAlarm
Probable Cause:	receiveFailure
Alarm Text:	Sonet Path STS AIS alarm.
Status Changes:	
1. Condition:	AIS condition is present on Path #n.
Alarm Status:	Critical
<text> Value:	AIS
Note:	The sonetPathCurrentStatus in the sonetPathCurrentTable has a value of sonetPathSTSAIS(4).
2. Condition:	AIS condition is not present.
Alarm Status:	Cleared

Table 4-31: acSonetPathSTSRDIAlarm

Alarm:	acSonetPathSTSRDIAlarm
OID:	1.3.6.1.4.1.5003.9.10.1.21.2.0.63
Default Severity:	Critical
Event Type:	communicationsAlarm
Probable Cause:	transmitFailure
Alarm Text:	Sonet Path STS RDI alarm.
Status Changes:	
1. Condition:	RDI condition is present on Path #n.
Alarm Status:	Critical
<text> Value:	RDI
Note:	The sonetPathCurrentStatus in the sonetPathCurrentTable has a value of sonetPathSTSRDI(8).
2. Condition:	RDI condition is not present.
Alarm Status:	Cleared

Table 4-32: acSonetPathUnequippedAlarm

Alarm:	acSonetPathUnequippedAlarm
OID:	1.3.6.1.4.1.5003.9.10.1.21.2.0.64
Default Severity:	Critical
Event Type:	communicationsAlarm
Probable Cause:	receiveFailure
Alarm Text:	Sonet Path Unequipped alarm.
Status Changes:	
1. Condition:	Unequipped condition is present on Path #n.
Alarm Status:	Critical
<text> Value:	Unequipped
Note:	The sonetPathCurrentStatus in the sonetPathCurrentTable has a value of sonetPathUnequipped(16).
2. Condition:	Unequipped condition is not present.
Alarm Status:	Cleared

Table 4-33: acSonetPathSignalLabelMismatchAlarm

Alarm:	acSonetPathSignalLabelMismatchAlarm
OID:	1.3.6.1.4.1.5003.9.10.1.21.2.0.65
Default Severity:	Critical
Event Type:	communicationsAlarm
Probable Cause:	receiveFailure
Alarm Text:	Sonet Path Signal Label Mismatch alarm.
Status Changes:	
1. Condition:	Signal Label Mismatch condition is present on Path #n.
Alarm Status:	Critical
<text> Value:	SignalLabelMismatch
Note:	The sonetPathCurrentStatus in the sonetPathCurrentTable has a value of sonetPathSignalLabelMismatch(32).
2. Condition:	Signal Label Mismatch condition is not present.
Alarm Status:	Cleared

Table 4-34: acSonetIfHwFailureAlarm

Alarm:	acSonetIfHwFailureAlarm
OID:	1.3.6.1.4.1.5003.9.10.1.21.2.0.42
Default Severity:	Critical on raise; Clear on clear
Event Type:	communicationsAlarm
Probable Cause:	Transmit failure
Alarm Text:	SONET/SDH IF Failure Alarm

4.19.2.5 DS3 Alarms

Note: These alarms are applicable only to Mediant 3000 with TP-6310 blade.

The source varbind text for the alarms under this component is Interfaces#0/DS3#<m>, where *m* is the DS3 IF number.

Table 4-35: acDS3RAIAlarm

Alarm:	acDS3RAIAlarm
OID:	1.3.6.1.4.1.5003.9.10.1.21.2.0.66
Default Severity:	Critical
Event Type:	communicationsAlarm
Probable Cause:	transmitFailure
Alarm Text:	DS3 RAI alarm.
Status Changes:	
1. Condition:	RAI condition is present on DS3-Line #n.
Alarm Status:	Critical
<text> Value:	RAI
Note:	The dsx3LineStatusfield in the dsx3ConfigTablewill have a value dsx3RcvRAIFailure(2).
2. Condition:	RIA condition is not present.
Alarm Status:	Cleared

Table 4-36: acDS3AISAlarm

Alarm:	acDS3AISAlarm
OID:	1.3.6.1.4.1.5003.9.10.1.21.2.0.67
Default Severity:	Critical
Event Type:	communicationsAlarm
Probable Cause:	receiveFailure
Alarm Text:	DS3 AIS alarm.
Status Changes:	
1. Condition:	AIS condition is present on DS3-Line #n.
Alarm Status:	Critical
<text> Value:	AIS
Note:	The dsx3LineStatusfield in the dsx3ConfigTablewill have a value dsx3RcvAIS(8).
2. Condition:	AIS condition is not present.
Alarm Status:	Cleared

Table 4-37: acDS3LOFAlarm

Alarm:	acDS3LOFAlarm
OID:	1.3.6.1.4.1.5003.9.10.1.21.2.0.68
Default Severity:	Critical
Event Type:	communicationsAlarm
Probable Cause:	lossOfFrame
Alarm Text:	DS3 LOF alarm.
Status Changes:	
1. Condition:	LOF condition is present on DS3-Line #n.
Alarm Status:	Critical
<text> Value:	LOF
Note:	The dsx3LineStatusfield in the dsx3ConfigTablewill have a value dsx3LOF (32).
2. Condition:	LOF condition is not present.
Alarm Status:	Cleared

Table 4-38: acDS3LOSAlarm

Alarm:	acDS3LOSAlarm
OID:	1.3.6.1.4.1.5003.9.10.1.21.2.0.69
Default Severity:	Critical
Event Type:	communicationsAlarm
Probable Cause:	lossOfSignal
Alarm Text:	DS3 LOS alarm.
Status Changes:	
1. Condition:	LOS condition is present on DS3-Line #n.
Alarm Status:	Critical
<text> Value:	LOS
Note:	The dsx3LineStatusfield in the dsx3ConfigTablewill have a value dsx3LOS (64).
2. Condition:	LOS condition is not present.
Alarm Status:	Cleared

Table 4-39: dsx3LineStatusChangeTrap

Alarm:	dsx3LineStatusChange
OID:	1.3.6.1.2.1.10.30.15.0.1
Default Severity:	Major on raise; Clear on clear
Event Type:	communicationsAlarm
Probable Cause:	A dsx3LineStatusChange trap is sent when the value of an instance of dsx3LineStatus changes. It can be utilized by an NMS to trigger polls. When the line status change results in a lower level line status change (i.e., ds1), then no traps for the lower level are sent.
Alarm Text:	DS3 Line Status
Additional Info1,2,3:	Updated DS3 Line Status. This variable indicates the Line Status of the interface. It contains loopback state information and failure state information. The dsx3LineStatus is a bit map represented as a sum, therefore it can represent multiple failures and a loopback (see dsx3LoopbackConfig object for the type of loopback) simultaneously. The dsx3NoAlarm must be set if and only if no other flag is set. If the dsx3loopbackState bit is set, the loopback in effect can be determined from the dsx3loopbackConfig object. The various bit positions are: 1 dsx3NoAlarm No alarm present 2 dsx3RcvRAIFailure Receiving Yellow/Remote Alarm Indication 4 dsx3XmitRAIAlarm Transmitting Yellow/Remote Alarm Indication 8 dsx3RcvAIS Receiving AIS failure state 16 dsx3XmitAIS Transmitting AIS 32 dsx3LOF Receiving LOF failure state 64 dsx3LOS Receiving LOS failure state 128 dsx3LoopbackState Looping the received signal 256 dsx3RcvTestCode Receiving a Test Pattern 512 dsx3OtherFailure Any line status not defined here 1024 dsx3UnavailSigState Near End in Unavailable Signal State 2048 dsx3NetEquipOOS Carrier Equipment Out of Service

4.19.2.6 Hitless Software Upgrade Alarms

Note: This alarm is applicable only to Mediant 3000.

Table 4-40: acHitlessUpdateStatus

Alarm:	acHitlessUpdateStatus
OID:	1.3.6.1.4.1.5003.9.10.1.21.2.0.48
Default Severity:	-
Event Type:	Other
Probable Cause:	Other
Alarm Text:	Hitless Update Event
Status Changes:	
Condition:	A Notification trap that is sent out at the beginning and the end of a Hitless SW update. Failure during the process will also instigate the trap. May include the following information: Hitless: start SW upgrade. Hitless: Stream read error, aborting CMP file processing. Hitless: Invalid cmp file - missing Ver parameter. Hitless fail: Hitless SW upgrade is not supported under version 5.2. Hitless fail: SW ver stream name too long. Hitless fail: Invalid cmp file - missing UPG parameter. Hitless fail: Hitless SW upgrade not supported. Hitless fail: Communication with redundant module failed. Hitless: SW upgrade ended successfully.
Alarm Status:	Indeterminate
Corrective Action:	

4.19.2.7 High Availability Alarms

Note: These alarms are applicable only to Mediant 3000 HA.

The source varbind text for the alarms under the component below is System#0/Module#<m>, where *m* is the blade module's slot number.

Table 4-41: acHASystemFaultAlarm

Trap:	acHASystemFaultAlarm
OID:	1.3.6.1.4.1.5003.9.10.1.21.2.0.33
Default Severity:	critical
Event Type:	qualityOfServiceAlarm
Probable Cause:	outOfService
Trap Text:	No HA! <text>
Status Changes:	
1. Condition:	HA feature is active but the system is not working in HA mode.
Trap Status:	Critical
<text> Value:	There are many possible values for the text: Fatal exception error TCPIP exception error Network processor exception error SW WD exception error HW WD exception error SAT device is missing SAT device error DSP error BIT tests error PSTN stack error Keep Alive error Software upgrade Manual switch over Manual reset Board removal Can't read slot number TER misplaced HW fault. TER in slot 2 or 3 is missing HW fault. TER has old version or is not functional HW fault. invalid TER Type HW fault. invalid TER active/redundant state HW fault. Error reading GbE state Redundant module is missing Unable to sync SW versions Redundant is not connecting Redundant is not reconnecting after deliberate restart No Ethernet Link in redundant module SA module faulty or missing
2. Condition:	HA feature is active and the redundant module is in start up mode and hasn't connected yet.
Trap Status:	Minor
<text> Value:	Waiting for redundant to connect
3. Condition:	HA system is active.
Trap Status:	Cleared

Table 4-42: acHASystemConfigMismatchAlarm

Trap:	acHASystemConfigMismatchAlarm
OID:	1.3.6.1.4.1.5003.9.10.1.21.2.0.34
Default Severity:	major
Event Type:	processingErrorAlarm
Probable Cause:	configurationOrCustomizationError
Trap Text:	Configuration mismatch in the system.
Status Changes:	
1. Condition:	HA feature is active: ▪ License Keys of Active and Redundant modules are different. ▪ The Active module was unable to pass on to the Redundant module the License Key. ▪ License key of the Redundant module is invalid.
Trap Status:	Major
<text> Value:	▪ Active and Redundant modules have different feature keys. ▪ Fail to update the redundant with feature key. ▪ Feature key did not update in redundant module.
2. Condition:	Successful License Key update.
Trap Status:	Cleared
<text> Value:	The feature key was successfully updated in the redundant module

Table 4-43: acHASystemSwitchOverAlarm

Trap:	acHASystemSwitchOverAlarm
OID:	1.3.6.1.4.1.5003.9.10.1.21.2.0.35
Default Severity:	Critical
Event Type:	qualityOfServiceAlarm
Probable Cause:	outOfService
Trap Text:	Switch-over: <text>
Status Changes:	
1. Condition:	A switch over from the active to the redundant blade has occurred.
Trap Status:	Critical
<text> Value:	See the acHASystemFaultAlarm table above.
2. Condition:	10 seconds have passed since the switch over.
Trap Status:	cleared

If the lost link is from the active module, the source varbind text for the alarm under this component is Chassis#0/Module#<m>/EthernetLink#0, where *m* is the blade's slot number.

Table 4-44: acBoardEthernetLinkAlarm

Trap:	acBoardEthernetLinkAlarm
OID:	1.3.6.1.4.1.5003.9.10.1.21.2.0.10
Default Severity:	Critical
Event Type:	equipmentAlarm
Probable Cause:	underlyingResourceUnavailable (56)
Trap Text:	Ethernet link alarm: <text>
Status Changes:	
1. Condition:	Fault on single interface of the Active module.
Trap Status:	Major
<text> Value:	Redundant link (physical link n) is down
2. Condition:	Fault on both interfaces
Trap Status:	Critical
<text> Value:	No Ethernet link
3. Condition:	Fault on single interface of the Redundant module.
Trap Status:	Major
<text> Value:	Redundant link in the redundant module (physical link n) is down
4. Condition:	Both interfaces are operational
Trap Status:	Cleared
Corrective Action:	Ensure that both Ethernet cables are plugged into the back of the system. Inspect the system's Ethernet link lights to determine which interface is failing. Reconnect the cable or fix the network problem
Note:	The alarm behaves differently when coming from the redundant or the active modules of an HA system. The alarm from the redundant is raised when there is an operational HA configuration in the system. There is no critical severity for the redundant module losing both its Ethernet links as that is conveyed in the no HA alarm that follows such a case.

4.19.2.8 Component: System#0<n> and Board#0<n>

The source varbind text for all the alarms under this component depends on the device:

- 3000 Series: **Board#0<n>**
- All other devices: **System#0<n>**

Where *n* is the slot number in which the blade resides in the chassis. For Mediant 1000 and MediaPack, *n* always equals to 1.

Table 4-45: acBoardFatalError

Alarm:	acBoardFatalError
OID:	1.3.6.1.4.1.5003.9.10.1.21.2.0.1
Default Severity:	Critical
Event Type:	equipmentAlarm
Probable Cause:	underlyingResourceUnavailable (56)
Alarm Text:	Board Fatal Error: <text>
Status Changes:	
1. Condition:	Any fatal error
Alarm Status:	Critical
<text> Value:	A run-time specific string describing the fatal error
2. Condition:	After fatal error
Alarm Status:	Status stays critical until reboot. A clear trap is not sent.
Corrective Action:	Capture the alarm information and the Syslog clause, if active. Contact your first-level support group. The support group will likely want to collect additional data from the device and perform a reset.

Table 4-46: acBoardConfigurationError

Alarm:	acBoardConfigurationError
OID:	1.3.6.1.4.1.5003.9.10.1.21.2.0.2
Default Severity:	Critical
Event Type:	equipmentAlarm
Probable Cause:	underlyingResourceUnavailable (56)
Alarm Text:	Board Config Error: <text>
Status Changes:	
1. Condition:	A configuration error was detected
Alarm Status:	critical
<text> Value:	A run-time specific string describing the configuration error.
2. Condition:	After configuration error
Alarm Status:	Status stays critical until reboot. A clear trap is not sent.
Corrective Action:	Inspect the run-time specific string to determine the nature of the configuration error. Fix the configuration error using the appropriate tool: Web interface, EMS, or *ini* file. Save the configuration and if necessary reset the device.

Table 4-47: acBoardTemperatureAlarm

Alarm:	acBoardTemperatureAlarm
OID:	1.3.6.1.4.1.5003.9.10.1.21.2.0.3
Default Severity:	Critical
Event Type:	equipmentAlarm
Probable Cause:	temperatureUnacceptable (50)
Alarm Text:	Board temperature too high
Status Changes:	
1. Condition:	Temperature is above 60°C (140°F)
Alarm Status:	Critical
2. Condition:	After raise, temperature falls below 55°C (131°F)
Alarm Status:	Cleared
Corrective Action:	Inspect the system. Determine if all fans in the system are properly operating.

Table 4-48: acBoardEvResettingBoard

Alarm:	acBoardEvResettingBoard
OID:	1.3.6.1.4.1.5003.9.10.1.21.2.0.5
Default Severity:	Critical
Event Type:	equipmentAlarm
Probable Cause:	outOfService (71)
Alarm Text:	User resetting board
Status Changes:	
1. Condition:	When a soft reset is triggered via the Web interface or SNMP.
Alarm Status:	Critical
2. Condition:	After raise
Alarm Status:	Status stays critical until reboot. A clear trap is not sent.
Corrective Action:	A network administrator has taken action to reset the device. No corrective action is required.

The trap below uses the following source varbind text:

- **All except 3000 Series:** Board#<n>/EthernetLink#0 (where *n* is the slot number)
- **3000 Series:** Module#<n>/EthernetLink#0 (where *n* is the slot number)

This trap relates to the Ethernet Link Module (the #0 numbering doesn't apply to the physical Ethernet link).

Table 4-49: acBoardEthernetLinkAlarm

Alarm:	acBoardEthernetLinkAlarm
OID:	1.3.6.1.4.1.5003.9.10.1.21.2.0.10
Default Severity:	Critical
Event Type:	equipmentAlarm
Probable Cause:	underlyingResourceUnavailable (56)
Alarm Text:	Ethernet link alarm: <text>
Status Changes:	
1. Condition:	Fault on single interface
Alarm Status:	Major
<text> Value:	Redundant link is down
2. Condition:	Fault on both interfaces
Alarm Status:	critical
<text> Value:	No Ethernet link
3. Condition:	Both interfaces are operational
Alarm Status:	cleared
Corrective Action:	Ensure that both Ethernet cables are plugged into the back of the system. Inspect the system's Ethernet link lights to determine which interface is failing. Reconnect the cable or fix the network problem

The source of the below alarm is Board#x/WanLink#y.

Table 4-50: acBoardWanLinkAlarm (Only for MSBG Devices)

Alarm:	acBoardWanLinkAlarm
OID:	1.3.6.1.4.1.5003.9.10.1.21.2.0.79
Default Severity:	Major / Clear
Event Type:	equipmentAlarm
Probable Cause:	underlyingResourceUnavailable
Alarm Text:	
Status Changes:	
1. Condition:	WAN link down
Alarm Status:	Major
<text> Value:	
2. Condition:	WAN link uo
Alarm Status:	Clear
<text> Value:	
Corrective Action:	

The source of the below alarm is Board#x/WanLink#y.

Table 4-51: acWirelessCellularModemAlarm (Only for Mediant 800 MSBG)

Alarm:	acWirelessCellularModemAlarm
OID:	1.3.6.1.4.1.5003.9.10.1.21.2.0.82
Default Severity:	Major / Clear
Event Type:	equipmentAlarm
Probable Cause:	underlyingResourceUnavailable
Alarm Text:	WAN wireless cellular modem alarm.
Status Changes:	
1. Condition:	Raised when either the wireless modem is down or in backup mode, and cleared when modem is up.
Alarm Status:	Major
2. Condition:	WAN link uo
Alarm Status:	Clear

Table 4-52: acBoardCallResourcesAlarm

Alarm:	acBoardCallResourcesAlarm
OID:	1.3.6.1.4.1.5003.9.10.1.21.2.0.8
Default Severity:	Major
Event Type:	processingErrorAlarm
Probable Cause:	softwareError (46)
Alarm Text:	Call resources alarm
Status Changes:	
1. Condition:	Percentage of busy channels exceeds the predefined RAI high threshold.
Alarm Status:	Major
Note:	To enable this alarm the RAI mechanism must be activated (EnableRAI = 1).
2. Condition:	Percentage of busy channels falls below the predefined RAI low threshold.
Alarm Status:	Cleared

Table 4-53: acBoardControllerFailureAlarm

Alarm:	acBoardControllerFailureAlarm
OID:	1.3.6.1.4.1.5003.9.10.1.21.2.0.9
Default Severity:	Minor
Event Type:	processingErrorAlarm
Probable Cause:	softwareError (46)
Alarm Text:	Controller failure alarm
Status Changes:	
1. Condition:	Proxy has not been found
Alarm Status:	Major
Additional Info:	Proxy not found. Use internal routing or Proxy lost. looking for another Proxy
2. Condition:	Proxy is found. The clear message includes the IP address of this Proxy.
Alarm Status:	Cleared

Table 4-54: acBoardOverloadAlarm

Alarm:	acBoardOverloadAlarm
OID:	1.3.6.1.4.1.5003.9.10.1.21.2.0.11
Default Severity:	Major
Event Type:	processingErrorAlarm
Probable Cause:	softwareError (46)
Alarm Text:	Board overload alarm
Status Changes:	
1. Condition:	An overload condition exists in one or more of the system components.
Alarm Status:	Major
2. Condition:	The overload condition passed
Alarm Status:	Cleared

Table 4-55: acFeatureKeyError

Alarm:	acFeatureKeyError
OID:	1.3.6.1.4.1.5003.9.10.1.21.2.0.6
Default Severity:	Critical
Event Type:	processingErrorAlarm
Probable Cause:	configurationOrCustomizationError (7)
Alarm Text:	Feature key error
Status Changes:	
Note:	Support for this alarm is pending.

The following trap is applicable only to the 3000 Series devices. The source varbind text for the alarm under this component is Chassis#0/SA#<m>, where *m* is the shelf Alarm module's slot number.

Table 4-56: acSAMissingAlarm (Applicable only to the 3000 Series Devices)

Alarm:	acSAMissingAlarm
OID:	1.3.6.1.4.1.5003.9.10.1.21.2.0.32
Default Severity:	Critical
Event Type:	equipmentAlarm
Probable Cause:	underlyingResourceUnavailable
Alarm Text:	SA Module Alarm. SA-Module from slot #n is missing.
Status Changes:	
1. Condition:	SA module removed or missing
Alarm Status:	Critical
2. Condition:	SA module is in slot 2 or 4 and working.
Alarm Status:	Cleared

Table 4-57: acNTPServerStatusAlarm

Alarm:	acNTPServerStatusAlarm
OID:	1.3.6.1.4.1.5003.9.10.1.21.2.0.71
Default Severity:	Major
Event Type:	communicationsAlarm
Probable Cause:	communicationsSubsystemFailure
Alarm Text:	NTP server alarm. No connection to NTP server.
Status Changes:	
1. Condition:	No initial communication to Network Time Protocol (NTP) server.
Alarm Status:	Major
2. Condition:	No communication to NTP server after the time was already set once.
Alarm Status:	Minor
Corrective Action:	Repair NTP communication. (The NTP server is down or its IP address is configured incorrectly in the device.)

Table 4-58: acNATTraversalAlarm

Alarm:	acNATTraversalAlarm
OID:	1.3.6.1.4.1.5003.9.10.1.21.2.0.17
Default Severity:	Indeterminate
Event Type:	-
Probable Cause:	other (0)
Alarm Text:	NAT Traversal Alarm
Status Changes:	The STUN client in the device is enabled and has either identified a NAT or is not finding the STUN server. Keep-alive is sent out every 9/10 of the time defined in the NatBindingDefaultTimeout parameter.
Corrective Action:	-

Table 4-59: acLDAPLostConnection

Alarm:	acLDAPLostConnection
OID:	1.3.6.1.4.1.5003.9.10.1.21.2.0.75
Default Severity:	Minor
Event Type:	communicationsAlarm
Probable Cause:	communicationsSubsystemFailure If a connection is idle for more than the maximum configured time in seconds that the client can be idle before the LDAP server closes the connection, the LDAP server returns an LDAP disconnect notification and this alarm is raised.
Alarm Text:	LDAP Lost Connection
Status Changes:	This alarm is raised when there is no connection to the LDAP server
1. Condition:	
Alarm Status:	

Table 4-60: acOCSPServerStatusAlarm

Alarm:	acOCSPServerStatusAlarm
OID:	1.3.6.1.4.1.5003.9.10.1.21.2.0.78
Default Severity:	Major / Clear
Event Type:	communicationsAlarm
Probable Cause:	communicationsSubsystemFailure
Alarm Text:	OCSP server alarm
Corrective Action	-

The trap below uses the following source varbind text: System#0/Interfaces#<n>.

Table 4-61: acIPv6ErrorAlarm (Applicable only to Mediant 800 E-SBC/3000 Series)

Alarm:	acIPv6ErrorAlarm
OID:	1.3.6.1.4.1.5003.9.10.1.21.2.0.53
Default Severity:	Critical
Event Type:	operationalViolation
Probable Cause:	communicationsProtocolError
Alarm Text:	IP interface alarm. <text>
Status Changes:	
1. Condition:	Bad IPv6 address (already exists)
Alarm Status:	Critical
<text> Value:	IPv6 Configuration failed, IPv6 will be disabled.
2. Condition:	After alarm raise
Alarm Status:	Status stays critical until reboot. A clear trap is not sent.
Corrective Action:	Find new IPV6 address and reboot.

Table 4-62: acgwAdminStateChange

Alarm:	acgwAdminStateChange
OID:	1.3.6.1.4.1.5003.9.10.1.21.2.0.7
Default Severity:	Major
Event Type:	processingErrorAlarm
Probable Cause:	outOfService (71)
Alarm Text:	Network element admin state change alarm Gateway is <text>
Status Changes:	
1. Condition:	Admin state changed to shutting down
Alarm Status:	Major
<text> Value:	shutting down. No time limit.
2. Condition:	Admin state changed to locked
Alarm Status:	Major
<text> Value:	locked
1. Condition:	Admin state changed to unlocked
Alarm Status:	cleared
Corrective Action:	A network administrator has taken an action to lock the device. No corrective action is required.

Table 4-63: acOperationalStateChange

Alarm:	acOperationalStateChange
OID:	1.3.6.1.4.1.5003.9.10.1.21.2.0.15
Default Severity:	Major
Event Type:	processingErrorAlarm
Probable Cause:	outOfService (71)
Alarm Text:	Network element operational state change alarm. Operational state is disabled.
Note:	This alarm is raised if the operational state of the node goes to disabled. The alarm is cleared when the operational state of the node goes to enabled.
Status Changes:	
1. Condition:	Operational state changed to disabled
Alarm Status:	Major
2. Condition:	Operational state changed to enabled
Alarm Status:	cleared
Note:	In IP systems, the operational state of the node is disabled if the device fails to properly initialize.
Corrective Action:	In IP systems, check for initialization errors. Look for other alarms and Syslogs that might provide additional information about the error.

Table 4-64: acSWUpgradeAlarm

Alarm:	acSWUpgradeAlarm
OID:	1.3.6.1.4.1.5003.9.10.1.21.2.0.70
Default Severity:	Major
Alarms Source:	System#0
Event Type:	processingErrorAlarm
Probable Cause:	softwareProgramError
Alarm Text:	SW upgrade error. <text>
Note:	
Condition:	Raised upon software upgrade errors.
Alarm Status:	major
<text> value:	Firmware burning failed. Startup system from Bootp/tftp.
Corrective Action:	Start up system from BootP/TFTP.

4.19.2.9 Alarm Manager Alarms

The source varbind text for all the alarms under this component is *System#0*<n>*/AlarmManager#0*.

Table 4-65: acActiveAlarmTableOverflow

Alarm:	acActiveAlarmTableOverflow
OID:	1.3.6.1.4.15003.9.10.1.21.2.0.12
Default Severity:	Major
Event Type:	processingErrorAlarm
Probable Cause:	resourceAtOrNearingCapacity (43)
Alarm Text:	Active alarm table overflow
Status Changes:	
1. Condition:	Too many alarms to fit in the active alarm table
Alarm Status:	Major
2. Condition:	After raise
Alarm Status:	Status remains Major until reboot. A Clear trap is not sent.
Note:	The status remains Major until reboot as it denotes a possible loss of information until the next reboot. If an alarm is raised when the table is full, it is possible that the alarm is active, but does not appear in the active alarm table.
Corrective Action:	Some alarm information may have been lost, but the ability of the device to perform its basic operations has not been impacted. A reboot is the only way to completely clear a problem with the active alarm table. Contact your first-level group.

4.19.2.10 Audio Staging Alarms

Note: This alarm is applicable only to IPmedia Series and Mediant 1000 devices.

The trap below uses the following source varbind text: System#0/AudioStaging#0

Table 4-66: acAudioProvisioningAlarm

Alarm:	acAudioProvisioningAlarm
OID:	1.3.6.1.4.1.5003.9.10.1.21.2.0.14
Default Severity:	Critical
Event Type:	processingErrorAlarm
Probable Cause:	configurationOrCustomizationError (7)
Alarm Text:	Unable to provision audio
Status Changes:	
1. Condition:	Media server times out waiting for a successful audio distribution from the APS (Audio Provisioning Server)
Alarm Status:	Critical
2. Condition:	After raise, media server is successfully provisioned with audio from the APS
Alarm Status:	Cleared
Corrective Action:	From the APS (Audio Provisioning Server) GUI ensure that the device is properly configured with audio and that the device has been enabled. Ensure that the IP address for the APS has been properly specified on the device. Ensure that both the APS server and application are in-service. For more information regarding the problem, view the Syslogs from the device as well as the APS manager logs.

4.19.2.11 Analog Port Alarms

Note: These alarms are applicable only to Analog devices.

The source varbind text for all the alarms under this component is System#0/analogports#<n>, where *n* is the port number.

Table 4-67: acAnalogPortSPIOutOfService

Alarm:	acAnalogPortSPIOutOfService
OID:	1.3.6.1.4.1.5003.9.10.1.21.2.0.46
Default Severity:	Major
Event Type:	physicalViolation
Probable Cause:	equipmentMalfunction
Alarm Text:	Analog Port SPI out of service
Status Changes:	
1. Condition:	Analog port has gone out of service
Alarm Status:	Major
2. Condition:	Analog port is back in service.
Alarm Status:	Cleared
Corrective Action:	None

Table 4-68: acAnalogPortHighTemperature

Alarm:	acAnalogPortHighTemperature
OID:	1.3.6.1.4.1.5003.9.10.1.21.2.0.47
Default Severity:	Major
Event Type:	physicalViolation
Probable Cause:	equipmentMalfunction
Alarm Text:	Analog Port High Temperature
Status Changes:	
1. Condition:	Analog device has reached critical temperature. Device is automatically disconnected.
Alarm Status:	Major
2. Condition:	Temperature is back to normal - analog port is back in service.
Alarm Status:	Cleared
Corrective Action:	None
Note:	Relevant to FXS only.

Table 4-69: acAnalogPortGroundFaultOutOfService

Alarm:	acAnalogPortGroundFaultOutOfService
OID:	1.3.6.1.4.1.5003.9.10.1.21.2.0.76
Default Severity:	Major / Clear
Event Type:	physicalViolation
Probable Cause:	equipmentMalfunction (This alarm is raised when the FXS port is inactive due to a ground fault)
Alarm Text:	Analog Port Ground Fault Out Of Service
Corrective Action:	-
Note:	Relevant to FXS only.

4.19.2.12 Media Alarm

Note: This alarm is applicable only to Mediant 800 MSBG, Mediant 800, Mediant 1000 MSBG, Mediant 1000B, Mediant 2000, and Mediant 3000.

Table 4-70: acMediaProcessOverloadAlarm

Alarm:	acMediaProcessOverloadAlarm
OID:	1.3.6.1.4.1.5003.9.10.1.21.2.0.81
Default Severity:	Major
Event Type:	environmentalAlarm
Probable Cause:	underlyingResourceUnavailable
Alarm Text:	Media Process Overload Alarm. %s
Status Changes:	
1. Condition:	
Alarm Status:	Major
2. Condition:	
Alarm Status:	Cleared
Corrective Action:	None

4.19.2.13 SIP Traps

Table 4-71: acGWSASEmergencyModeAlarm

Alarm:	acGWSASEmergencyModeAlarm
OID:	1.3.6.1.4.1.5003.9.10.1.21.2.0.59
Default Severity:	
Event Type:	Other
Probable Cause:	Other
Alarm Text:	-
Status Changes:	Sent by the Stand-Alone Survivability (SAS) application when switching from “Normal" mode to “Emergency" mode. This alarm is cleared once the SAS returns to “Normal" mode.
Corrective Action:	-

4.19.3 Log Traps (Notifications)

This subsection details traps that are not alarms. These traps are sent with the severity varbind value of ‘indeterminate’. These traps don’t ‘clear’ and they don’t appear in the alarm history or active tables. (The only log trap that does send clear is acPerformanceMonitoringThresholdCrossing.)

Table 4-72: acPowerOverEthernetStatus (Applicable only to Mediant 800 MSBG)

Trap:	acPowerOverEthernetStatus
OID:	1.3.6.1.4.1.5003.9.10.1.21.2.0.80
Default Severity:	Indeterminate
Event Type:	environmentalAlarm
Probable Cause:	underlyingResourceUnavailable
Trap Text:	“POE Port %d Was Not Powered Due To Power Management” Where, %d is the Ethernet port number.
Condition:	This trap is sent when insufficient power is available for a plugged-in PoE client in a PoE-enabled LAN port.
Trap Status:	Trap is sent

Table 4-73: acKeepAlive

Trap:	acKeepAlive
OID:	1.3.6.1.4.1.5003.9.10.1.21.2.0.16
Default Severity:	Indeterminate
Event Type:	other (0)
Probable Cause:	other (0)
Trap Text:	Keep alive trap
Status Changes:	
Condition:	The STUN client in is enabled and identified a NAT device or doesn't locate the STUN server. The *ini* file contains the following line: 'SendKeepAliveTrap=1'
Trap Status:	Trap is sent
Note:	Keep-alive is sent every 9/10 of the time defined in the parameter NatBindingDefaultTimeout.

Table 4-74: acPerformanceMonitoringThresholdCrossing

Trap:	acPerformanceMonitoringThresholdCrossing
OID:	1.3.6.1.4.1.5003.9.10.1.21.2.0.27
Default Severity:	Indeterminate
Event Type:	other (0)
Probable Cause:	other (0)
Trap Text:	"Performance: Threshold trap was set", with source = name of performance counter which caused the trap
Status Changes:	
Condition:	A performance counter has crossed the high threshold
Trap Status:	Indeterminate
Condition:	A performance counter has returned to under the threshold
Trap Status:	Cleared

Table 4-75: acHTTPDownloadResult

Trap:	acHTTPDownloadResult
OID:	1.3.6.1.4.1.5003.9.10.1.21.2.0.28
Default Severity:	Indeterminate
Event Type:	processingErrorAlarm (3) for failures and other (0) for success.
Probable Cause:	other (0)
Status Changes:	
Condition:	Successful HTTP download.
Trap Text:	HTTP Download successful
Condition:	Failed download.
Trap Text:	HTTP download failed, a network error occurred.
Note:	There are other possible textual messages describing NFS failures or success, FTP failure or success.

Table 4-76: acDialPlanFileReplaced (Applicable Only to Digital PSTN)

Alarm:	acDialPlanFileReplaced
OID:	1.3.6.1.4.1.5003.9.10.1.21.2.0.45
Default Severity:	Indeterminate
Event Type:	Other (0)
Probable Cause:	Other (0)
Status Change:	
Condition:	Successful dial plan file replacement
Trap Text:	Dial plan file replacement complete.

Table 4-77: acHitlessUpdateStatus (Applicable Only to 3000 Series)

Alarm:	acHitlessUpdateStatus
OID:	1.3.6.1.4.1.5003.9.10.1.21.2.0.48
Default Severity:	Indeterminate
Event Type:	Other (0)
Probable Cause:	Other (0)
Source:	Automatic Update
Status Changes:	
Condition:	Successful SW upgrade
Trap Text:	Hitless: SW upgrade ended successfully
Condition:	Failed SW upgrade
Trap Text:	Hitless fail: Waiting for module in slot <n> to burn new SW and reboot Timed out. (n – slot number).

Table 4-78: acSSHConnectionStatus

Alarm:	acSSHConnectionStatus
OID:	1.3.6.1.4.1.5003.9.10.1.21.2.0.77
Default Severity:	indeterminate
Event Type:	environmentalAlarm
Probable Cause:	other
Alarm Text:	"SSH successful login from IP address %s, user %s" "SSH unsuccessful login attempt from IP address %s, user %s"
Status Changes:	
Condition:	SSH connection attempt
<text> Value:	%s – remote IP %s – user name
Condition:	SSH connection attempt – success of failure

4.19.4 Other Traps

The following are provided as SNMP traps and are not alarms.

Table 4-79: coldStart

Trap Name:	coldStart
OID:	1.3.6.1.6.3.1.1.5.1
MIB:	SNMPv2-MIB
Note:	This is a trap from the standard SNMP MIB.

Table 4-80: authenticationFailure

Trap Name:	authenticationFailure
OID:	1.3.6.1.6.3.1.1.5.5
MIB:	SNMPv2-MIB

Table 4-81: acBoardEvBoardStarted

Trap Name:	acBoardEvBoardStarted
OID:	1.3.6.1.4.1.5003.9.10.1.21.2.0.4
MIB:	AcBoard
Severity:	cleared
Event Type:	equipmentAlarm
Probable Cause:	Other(0)
Alarm Text:	Initialization Ended
Note:	This is the AudioCodes Enterprise application cold start trap.

Table 4-82: entConfigChange

Trap Name:	entConfigChange
OID:	1.3.6.1.2.1.4.7.2
MIB:	ENTITY-MIB

Table 4-83: linkUp

Trap Name:	linkUp
OID:	1.3.6.1.6.3.1.1.5.4
MIB:	IF-MIB

Table 4-84: linkDown

Trap Name:	linkDown
OID:	1.3.6.1.6.3.1.1.5.3
MIB:	IF-MIB

Table 4-85: AcDChannelStatus (Applicable Only to Digital PSTN)

Trap Name:	acDChannelStatus
OID:	1.3.6.1.4.1.5003.9.10.1.21.2.0.37
MIB	AcBoard
Severity:	Minor
Event Type:	communicationsAlarm
Probable Cause:	communicationsProtocolError
Alarm Text:	D-Channel Trap.
Source:	Trunk <m> where m is the trunk number (starts from 0).
Status Changes:	
Condition:	D-Channel un-established.
Trap Status:	Trap is sent with the severity of Minor.
Condition:	D-Channel established.
Trap Status:	Trap is sent with the severity of Cleared.

4.19.5 Trap Varbinds

Each trap described above provides the following fields (known as *varbinds*). Refer to the AcBoard MIB for additional details on these varbinds.

- acBoardTrapGlobalsName
- acBoardTrapGlobalsTextualDescription
- acBoardTrapGlobalsSource
- acBoardTrapGlobalsSeverity
- acBoardTrapGlobalsUniqID
- acBoardTrapGlobalsType
- acBoardTrapGlobalsProbableCause
- acBoardTrapGlobalsDateAndTime
- acBoardTrapGlobalsAdditionalInfo1
- acBoardTrapGlobalsAdditionalInfo2
- acBoardTrapGlobalsAdditionalInfo3

Note: 'acBoardTrapGlobalsName' is actually a number. The value of this varbind is 'X' minus 1, where 'X' is the last number in the trap's OID. For example, the 'name' of 'acBoardEthernetLinkAlarm' is '9'. The OID for 'acBoardEthernetLinkAlarm' is 1.3.6.1.4.1.5003. 9.10.1.21.2.0.10.

4.19.6 Customizing Trap's Enterprise OID

You can change the enterprise value in the device's SNMP Traps to a variable value using the *ini* parameter SNMPTrapEnterpriseOid. This parameter replaces the Traps' OID prefix from 'AcTrap' (1.3.6.1.4.1.5003.9.10.1.21) to user-defined root. All other OIDs remain the same.

For example, the current acBoardEvBoardStarted parameter's OID is '1.3.6.1.4.1.5003.9.10.1.21.2.0.4'. Its prefix ('1.3.6.1.4.1.5003.9.10.1.21') can be changed, and all other OIDs remain the same.

4.20 Getting Started with SNMP

This section provides a getting started for quickly setting up the device for management using AudioCodes SNMP MIBs.

4.20.1 Basic SNMP Configuration Setup

This subsection provides a description of the required SNMP configuration when first accessing the SNMP agent running on the device.

To access the device's SNMP agent, there are a few parameters that can be configured if you wish not to use default settings. The SNMP agent default settings include the following:

- SNMP agent is enabled.
- Port 161 in the agent is used for SNMP GET/SET commands.
- No default trap managers are defined, therefore, the device does not send traps.
- The Trap destination port is 162.
- The SNMP agent is accessible to all SNMP managers (i.e., no trusted managers).
- SNMP Protocol version - SNMPv2c with 'public' and 'private' as the read-only and read-write community strings respectively.

Configuring these SNMP attributes is described in the following subsections:

4.20.1.1 Disabling SNMP

To disable SNMP, in the ini file parameter set the following:

```
DisableSNMP = 1
```

4.20.1.2 Configuring SNMP Port

To configure the agent's SNMP port in the ini file, set the following

```
SNMPPort = <x>
; where 'x' is the port number.
```

4.20.1.3 Configuring Trap Mangers (Trap Destination)

Configuring Trap Managers (i.e., trap destinations) includes defining IP address and port. This configuration corresponds to the snmpTargetAddrTable. The agent supports up to five separate trap destinations. For each manager, you need to set the manager IP address and trap-receiving port along with enabling the sending to that manager. Trap managers can be configured using ini file, SNMP, or Web interface.

In addition, you can associate a trap destination with a specific SNMPv3 USM user. Traps will be sent to that trap destination using the SNMPv3 format and the authentication and privacy protocol configured for that user.

- **Using ini File:** two options that can be used separately or together:
  - Explicit IP address:

```
SNMPMANAGERTABLEIP_x=<IP address>
SNMPMANAGERISUSED_x=1
SNMPMANAGERTRAPSENDINGENABLE_x=1
SNMPMANAGERTRAPPORT_x=162 ;(optional)
```

Where *x* is the entry index from 0 to 4.

  - Manager host name:

```
SNMPTrapManagerHostName = <’host name on network’>
```

For example: 'myMananger.corp.MyCompany.com’

The host name is translated into the IP address using DNS resolution and is then defined as the fifth (last) trap manager. Until the address is resolved, some traps are expected to be lost.

Notes:

- This option also requires you to configure the DNS server IP address (in the Multiple Interface table).
- This option results in the fifth manager being overrun by the resolved IP address. Online changes to the Manager table will also be overrun.

- **Using SNMP:** The trap managers are SET using the SNMPTargetMIB MIB onbject.
  - To add an SNMPv2 trap destination: Add a row to the snmpTargetAddrTable with these values:
    - Name=trapN, where *N* is an unused number between 0 and 4.
    - TagList=AC_TRAP
    - Params=v2cparamsm

    All changes to the trap destination configuration take effect immediately.
  - To add an SNMPv3 trap destination:
    1. Add a row to the snmpTargetAddrTable with these values: Name=trapN, >, where *N* is an unused number between 0 and 4, and <*user*> is the name of the SNMPv3 that this user is associated with:
       - ✓ TagList=AC_TRAP

- ✓ Params=usm<user>

2. If a row does not already exist for this combination of user and SecurityLevel, add a row to the snmpTargetParamsTable with this values:
   - ✓ Name=usm<user>
   - ✓ MPModel=3(SNMPv3)
   - ✓ SecurityModel=3 (usm)
   - ✓ SecurityName=<user>
   - ✓ SecurityLevel=M, where *M* is either 1(noAuthNoPriv), 2(authNoPriv) or 3(authPriv)

- To delete a trap destination:
  1. Remove the appropriate row from the snmpTargetAddrTable.
  2. If this is the last trap destination associated with this user and security level, you can also delete the appropriate row from the snmpTargetParamsTable.
- To modify a trap destination, change the IP address and or port number for the appropriate row in the snmpTargetAddrTable for an existing trap destination. The same effect can be achieved by removing a row and adding a new row.
- To disable a trap destination, change TagList on the appropriate row in the snmpTargetAddrTable to the empty string.
- To enable a trap destination, change TagList on the appropriate row in the snmpTargetAddrTable to "AC_TRAP".

- **Using Web Interface:** The Trap Destination table appears in the ‘SNMP Trap Destinations’ page (**Configuration** tab > **System** menu > **Management** > **SNMP** > **SNMP Trap Destinations**). The check box on the left indicates if the row is used. The three columns are used to set IP address, port and enable trap sending. The SNMPv3 Settings table, also accessed from the ‘Management Setting’ page is used for setting trap users.
  - To add a trap user: In the field near the **Add Index** button, enter the index of the row you want to add (0 to 9), and then click the button. The row is now available for configuration. The five columns include name, authentication protocol, privacy protocol, authentication key and privacy key. After configuring the columns, click **Apply**.
  - To delete a row: Select the corresponding index field, and then click **Delete**.

4.20.1.4 Configuring Trap Destination Port

For configuring the trap destination port, see trap managers, above.

4.20.1.5 Configuring Trusted Managers

The configuration of trusted managers determines which managers can access the device. You can define up to five trusted managers.

Notes:

- The concept of trusted managers is considered to be a weak form of security and is therefore, not a required part of SNMPv3 security, which uses authentication and privacy.
- Trusted managers are therefore, not supported in SNMPv3 – thus they apply only when the device is set to use SNMPv2c.
- If trusted managers are defined, then all community strings work from all trusted managers. That is, there is no way to associate a community string with particular trusted managers.

The configuration can be done via ini file, SNMP and Web.

- **Using ini file:** SNMPTRUSTEDMGR_x = <IP address>, where *x* is the entry index 0 to 4.
- **Using SNMP:** To configure Trusted Managers, the EM must use the SNMP-COMMUNITY-MIB, snmpCommunityMIB, and snmpTargetMIB.
  - To add the first Trusted Manager: This procedure assumes that there is at least one configured read-write community. There are currently no Trusted Managers. The TransportTag for columns for all snmpCommunityTable rows are currently empty.
    1. Add a row to the snmpTargetAddrTable with these values:
       - ✓ Name=mgr0
       - ✓ TagList=MGR
       - ✓ Params=v2cparams.
    2. Add a row to the snmpTargetAddrExtTable table with these values:
       - ✓ Name=mgr0
       - ✓ snmpTargetAddrTMask=255.255.255.255:0.

       The agent does not allow creation of a row in this table unless a corresponding row exists in the snmpTargetAddrTable.
    3. Set the value of the TransportTag field on each non-TrapGroup row in the snmpCommunityTable to MGR.
  - To add a subsequent Trusted Manager: This procedure assumes that there is at least one configured read-write community. There are currently one or more Trusted Managers. The TransportTag for columns for all rows in the snmpCommunityTable are currently set to MGR. This procedure must be done from one of the existing Trusted Managers.
    1. Add a row to the snmpTargetAddrTable with these values:
       - ✓ Name=mgrN, where N is an unused number between 0 and 4.
       - ✓ TagList=MGR
       - ✓ Params=v2cparams

2. Add a row to the snmpTargetAddrExtTable table with these values:
   - ✓ Name=mgrN
   - ✓ snmpTargetAddrTMask=255.255.255.255:0.

   An alternative to the above procedure is to set the snmpTargetAddrTMask column while you are creating other rows in the table.

- To delete a Trusted Manager (not the final one): This procedure assumes that there is at least one configured read-write community. There are currently two or more Trusted Managers. The taglist for columns for all rows in the snmpCommunityTable are currently set to MGR. This procedure must be done from one of the existing trusted managers, but not the one that is being deleted. Remove the appropriate row from the snmpTargetAddrTable; The change takes effect immediately. The deleted trusted manager cannot access the device. The agent automatically removes the row in the snmpTargetAddrExtTable.
- To delete the final Trusted Manager: This procedure assumes that there is at least one configured read-write community. There is currently only one Trusted Manager. The taglist for columns for all rows in the snmpCommunityTable are currently set to MGR. This procedure must be done from the final Trusted Manager.
  1. Set the value of the TransportTag field on each row in the snmpCommunityTable to the empty string.
  2. Remove the appropriate row from the snmpTargetAddrTable; The change takes effect immediately. All managers can now access the device. The agent automatically removes the row in the snmpTargetAddrExtTable.
- Using Web interface: Under the **Configuration** tab, choose **System**, **Management**, **SNMP**, and then click **SNMP Trusted Managers**. The Web now displays the table. Use the **Submit** button for applying your configuration. Use the check boxes for deleting.

4.20.2 Familiarizing Yourself with AudioCodes MIBs

Audiocodes' proprietary MIBs are located in the AudioCodes subtree (OID 1.3.6.1.4.1.5003). A classification within the subtree separates the MIBs according to the following:

- Configuration and status MIBs – in the acBoardMibs subtree
- Performance monitoring MIBs – in the acPerformance subtree
- Proprietary Carrier Grade Alarm MIB – in the acFault subtree

In the acBoardsMibs and acPerformance subtrees, the different MIB modules are grouped according to different virtual modules of AudioCodes' devices. In general, the division is as follows (a more detailed breakdown of the MIBs is discussed below):

- **acBoardMibs subtrees:**
  - **acBoard MIB:** proprietary traps.
  - **acGateway MIB:** SIP control protocol specific objects. This MIB is supported only in SIP devices. This MIB's structure is unlike the other configuration and status MIBs.
  - **acMedia MIB:** DSP and media related objects. This MIB includes the configuration and status of DSP, voice, modem, fax, RTP/RTCP related objects. This MIB is relevant to all devices.

- **acControl MIB:** mostly MEGACO and MGCP CP related objects. A number of objects are also related to SIP. The MIB is divided into subtrees that are common to both MEGACO and MGCP (amongst these are also the SIP relevant objects) and subtrees that are specific to the different CPs. This MIB is relevant to all devices.
- **acAnalog MIB:** all objects in this MIB are related only to the configuration, status and line testing or resetting of analog interfaces. This MIB is relevant to devices with analog interfaces only.
- **acPSTN MIB:** configuration and status of trunk related objects only. Most of the MIB objects are trunk specific. This MIB is relevant to devices with digital PSTN interfaces only.
- **acSystem MIB:** configuration and status of a wide range of general objects along with chassis related objects and a variety of actions that can be instigated. The MIB is relevant to all devices.
- **acV5 MIB:** configuration and status of v5.2 related objects only. This MIB is relevant to Mediant 3000/TP-6310.

- **acPerformance subtrees:**
  - acPMMedia, acPMControl, acPMAnalog, acPMPSTN, acPMSystem: module specific parameters performance monitoring MIBs
  - acPMMediaServer MIB: performance monitoring specifically for MediaServer related parameters (IVR, BCT, Conference and Trunk-Testing)
  - acPerfH323SIPGateway MIB: performance specific for SIP CP devices. This MIB's structure is unlike the other performance monitoring MIBs.
- **acFault subtree:** only one MIB exists – the acAlarm which is a proprietary simplification of the standard notificationLogMIB and alarmMIB (both are also supported).

The structure of the different MIBs is similar, depending on the subtree in which they reside. The MIBs in the acBoardMibs subtree have a very similar structure (except the acBoard and acGateway MIBs). Each MIB can be made up of four major subtrees:

- **Configuration subtree:** mostly read-write objects, tables and scalars. The relevant module's configuration is done via these objects.
- **Status subtree:** read-only objects, tables and scalars. Module status is collected by these objects.
- **Action subtree:** read-write objects that are used to instigate actions on the device (such as reset, save configuration, and so on) and read-only objects used to receive the actions' results.
- **Chassis subtree (in acSystem MIB only):** read-write and read-only objects related to chassis control and management (this includes, fan trays, power supply modules, PSTN IF modules, etc').

The acBoard MIB contains some deprecated objects and current proprietary trap definitions.

The acGateway MIB contains only the configuration subtree which in return is divided into common, SIP and H323 subtrees. The H323 subtree is mostly deprecated or obsolete.

4.20.3 Performance Monitoring Overview

Performance monitoring (PM) are available for a Third-Party Performance Monitoring System through an SNMP interface and can be polled at any interval by an external poller or utility in the management server or other off device system.

This section describes AudioCodes proprietary performance measurements (PM) MIB.

The device's performance measurements are provided by several proprietary MIBs (located under the "acPerformance" subtree (see below for more detail on each of the MIBs):

- **acPMMedia:** for media (voice) related monitoring such as RTP and DSP.
- **acPMControl:** for Control Protocol related monitoring such as connections, commands.
- **acPMAnalog:** Analog channels off-hook state (applicable to devices with analog interfaces only)
- **acPMPSTN:** for PSTN related monitoring such as channel use, trunk utilization.
- **cPMSystem:** for general (system related) monitoring.
- **acPMMediaServer:** for Media Server specific monitoring. (Applicable to the 3000/6310/8410 devices)

Performance Monitoring MIBs have a fixed format. They all have an identical structure consisting of two major subtrees:

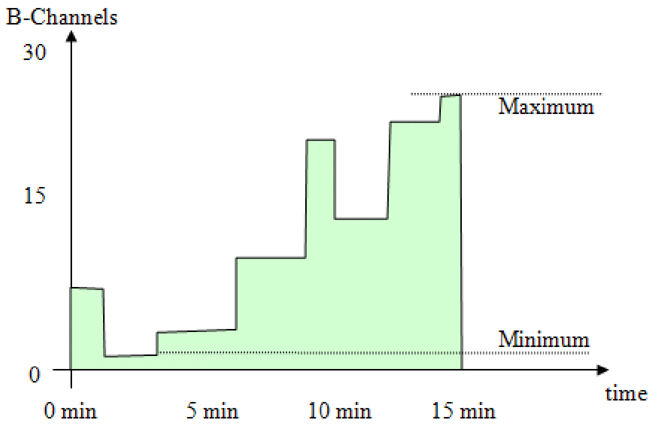
- **Configuration subtree:** allows configuration of general attributes of the MIB and specific attributes of the monitored objects.
- **Data subtree:** this is where the monitored information is found.

The information supplied by the device is divided into time intervals (default is 15 minutes). These intervals are used as a key in the tables. Thus, the monitoring results are presented in tables. There are one or two indices in each table. If there are two, the first is a sub-set in the table (e.g., trunk number) and the second (or the single where there is only one) index represents the interval number (present - 0, previous - 1 and the one before - 2).

Some of the PM parameters support a history with more than two intervals. These include the MEGACO parameters, IVR requests, IVR-play-collect, IVR-play-record, BCT contexts, conference calls, trunk-test calls and digit-collect requests.

Note: The interval's start time is synchronized with the device's clock so that they begin on the hour. If you are using NTP, then it is likely that the last interval within the first hour after device start up will be cut short to accommodate for this synchronization.

Following is a graphic example of one monitored parameter, in this case the number of utilized B-channels in a single trunk:



The x-axis is the time within the interval. The y-axis is the number of used channels. The parameter’s value is a gauge. While the interval index is 0 (thus it is the current interval, any GET on the parameter value will return y-axis value for the graph at that moment in time. When the interval is over (index 1 or 2) the value is no longer relevant but there are other attributes such as the average – in this case the area in green divided by the interval length in seconds.

The configuration subtree includes:

- **Reset Total Counters:** resets the ‘total’ (see below) objects in all the MIB’s tables if they are defined.
- **Attributes subtrees:** a number of subtrees in which scalars are used to configure the high and low thresholds for relevant tables.

The Data subtree consists of monitored data and statistics:

- **Time From Start Of Interval object:** GETs the time in seconds from the beginning of the current interval.
- **Data tables:** all have similar structure. Not all possible columns appear in all of them. The specific structure of a table (i.e. what columns are defined) is parameter specific. The only column that always appears is the interval column. The information in each column is a statistical attribute of the parameter being looked at.

Note: When an attribute value is -1, it means that the attribute isn’t relevant at that point of time.

The columns are:

- Table specific index – table key.
- Interval – index, 0,1,2 – table key.
- Val – value of gauge or counter. This is the snapshot view of current device activity.
  - Counter – cumulative, only increases in value.
  - Gauge – fluctuates in value, value increases and decreases.
- Average – within the period length.
- Max – gauge high water mark.
- Min - gauge low water mark.

- Volume – number of times gauge or counter was updated, indicating the volume of change. For example:
  - For a trunk utilization element, the volume indicates how many calls were made and released.
  - For the Ethernet connection status element, the volume indicates how many network connections and disconnections occurred.
- TimeBelowLowThreshod – Percent of interval time for which the gauge is below the determined low threshold.
- TimeAboveHighThreshod – Percent of interval time for which the gauge is above the determined high threshold.
- TimeBetweenThresholds – Percent of interval time for which the gauge is between thresholds.
- FullDayAverage – 24 hour average.
- Total – relevant when using counters. Sums all counter values so far. It resets only once every 24 hours.
- StateChanges – the number of times a state (mostly active/non-active) was toggled.

The log trap, acPerformanceMonitoringThresholdCrossing (non-alarm) is sent out every time the threshold of a Performance Monitored object is crossed. The severity field is 'indeterminate' when the crossing is above the threshold and 'cleared' when it returns to under the threshold. The 'source' varbind in the trap indicates the object for which the threshold is being crossed.

Expansions for the different MIBs.

- **acPMMedia:** Consists of data related to voice, DSPs coders etc. This MIB includes the following parameters:
  - Number of active DSP channels
  - Channels used for each coder
  - Discarded packets in robust RTP filter
  - Media Networking subtree - an array of packet behavior parameters such as delay, jitter, transmitted/received and lost RTP bytes and packets.
  - Media Networking Aggregated subtree - displays similar data only for the entire device and includes TDM-IP and IP-IP calls.
  - Channel Utilization subtree - parameters regarding channel use by fax, modem, TDM-IP calls, RTP, SRTP, multicast source and modem relay.
  - Streaming Cache subtree - hit count, miss count and server request count.
- **acPMControl:** Control Protocol related monitoring is divided into three groups – MEGACO, MGCP and SIP. The MIB includes the following parameters:
  - CP Connection subtree – general for all three control protocols. Its parameters include connection lifetime/state, counters for commands, retransmissions, active contexts, command success/failure and process time, transaction processing time and call attempts.
  - The remaining three subtrees are self explanatory and are CP specific.
- **acPMAnalog:** Analog channels statistics - one table only (offhook state).
- **acPMPSTN:** All statistics in this MIB are per trunk:
  - Number of active channels.

  - Trunk activity.
  - Number of channels that are in/out of service and in maintenance.

- **acPMSystem:** This detailed MIB is for general (system related) monitoring:
  - IP connection.
  - Discarded UDP packets due to unknown port.
  - System Net Utils subtree – transmitted/received bytes/packets, discarded packets.
  - System Network subtree – DHCP response time/request count. STUN related statistics.
  - System Sctp subtree – SCTP sent/received/retransmitted bytes, retransmission attempts.
  - IPsec security associations.
  - System Multicast subtree – multicast IP packets received, multicast IP packets conveying UDP payload packets received/rejected, IGMP packets/general-queries/specific-queries received, IGMP membership-report/leave-group sent messages.
  - System Congestion subtree – congestion state for general resources, DSP resources, IP resources, conference resources. (ATM resources table is obsolete).
  - System NFS subtree – NFS related parameters.
  - System MSBG subtree – includes received good/bad octets, received undersized/oversized/discarded packets, received MAC errors, received FSC error packets, transmitted octets/packets/collisions/late-packets.
- **acPMMediaServer:** (Applicable to the 3000/6310/8410 devices) The Media Server related data is divided into four subtrees:
  - IVR subtree – play requests, play progress/duration/collect/collect-in-progress/collect-duration/record/record-in-progress/record-duration, digit-collect requests, digit-collect in-progress/duration.
  - BCT subtree – BCT contexts, BCT in-progress/duration.
  - Conference subtree – conference calls, conference in-progress/duration.
  - Trunk Test subtree – trunk test requested, trunk tests in-progress/duration.

4.20.4 Traps and Alarms

AudioCodes supports standard traps and proprietary traps. Most of the proprietary traps are alarm traps, that is, they can be raised and cleared. Thus, they are referred to as *alarm traps*. All the standard traps are non-alarm traps, referred to as *log traps*. The complete list of all supported traps is mentioned in previous subsections.

The proprietary traps are defined under the acBoardTrapDefinitions subtree.

The standard MIB traps supported include the following:

- coldStart
- authenticationFailure
- linkDown
- linkup
- dsx1LineStatusChange
- rtcpXrVoipThresholdViolation
- dsx3LineStatusChange
- entConfigChange

This subsection describes the device’s configuration so that traps are sent out to user-defined managers under SNMPv2c or SNMPv3. It continues with an explanation on the ‘carrier grade alarm’ abilities and usage.

4.20.4.1 Device Configuration

For a device to send out traps to specified managers the most basic configuration are the trap targets. More advanced configuration includes the Trap Community String or traps over SNMPv3.

- Destination IP address and port (see Basic SNMP Configuration Setup)
- Trap Community String: The default Trap Community String is ‘trapuser’. There is only 1 for the entire device. It can be configured via ini file, SNMP or Web:
    - INI file: SNMPTRAPCOMMUNITYSTRING = <your community string here>.
    - SNMP: add a new community string to the snmpCommunityTable. To associate the traps to the new Community String change the snmpTargetParamsSecurityName in the snmpTargetParamsTable so it coincides with the snmpCommunitySecurityName object. If you wish, you can remove the older Trap Community String from snmpCommunityTable (however, it is not mandatory).
    - Web: under the ‘Management’ tab, choose ‘Management Settings’ in the ‘Management Settings’ menu. On the page, click the **SNMP Community String** arrow to display the table. Use the **Submit** button to apply your configuration. You can’t delete the Trap Community String, only modify its value.

- SNMPv3 Settings: When using SNMPv3 settings it is important to note that by default the trap configuration remains such that the traps are sent out in SNMPv2c mode. To have traps sent out in SNMPv3, you can use either ini file or SNMP:

  - INI file: amongst the SNMPv3 users ensure that you also define a trap user (the value of 2 in the SNMPUsers_Group indicates the trap user). For example: you can have the SNMP users table defined with a read-write user, ‘rwmd5des’ with MD5 authentication and DES privacy, along with a trap user, ‘tmd5no’ with SHA authentication and DES privacy:

```
[ SNMPUsers ]
FORMAT SNMPUsers_Index = SNMPUsers_Username,
SNMPUsers_AuthProtocol, SNMPUsers_PrivProtocol,
SNMPUsers_AuthKey, SNMPUsers_PrivKey, SNMPUsers_Group;
SNMPUsers 1 = rwmd5des, 1, 1, myauthkey, myprivkey, 1;
SNMPUsers 2 = tshades, 2, 1, myauthkey, myprivkey, 2
[ \SNMPUsers ]
```

Notes:

- If you define a trap user only, the device runs in SNMPv3 mode but will not be accessible as there are no defined read-write or even read-only users.
- If you define non-default community strings (SNMPv2c), you need to access the device via SNMPv2c.

Along with this configuration, you also need to associate the trap targets (managers) with the user:

```
SNMPMANAGERTRAPUSER_x=tshades
```

where *x* is the target index and can be between 0 and 4.

Any targets that are defined in the ini file where this last parameter isn’t defined, receives SNMPv2c traps.

  - SNMP: change snmpTargetAddrParams object to the user of your choice adding the letters ‘usm’ as prefix (ensure it’s a trap user). For example, the ‘tshades’ user should be added as ‘usmtshades’.

4.20.4.2 Carrier Grade Alarm (CGA)

A carrier-grade alarm system provides a reliable alarm reporting mechanism that takes into account element management system outages, network outages, and transport mechanism such as SNMP over UDP.

A carrier-grade alarm system is characterized by the following:

- The device allows a manager to determine which alarms are currently active in the device. That is, the device maintains an active alarm table.
- The device allows a manager to detect lost alarms and clear notifications (sequence number in trap, current sequence number MIB object).
- The device allows a manager to recover lost alarm raise and clear notifications (maintains a log history).
- The device sends a cold start trap to indicate that it is starting. This allows the manager to synchronize its view of the device's active alarms.

When the SNMP alarm traps are sent, the carrier-grade alarm system does not add or delete alarm traps as part of the feature. This system provides the mechanism for viewing history and current active alarm information.

As part of CGA, the device supports the following:

- **Active Alarm Table:** The device maintains an active alarm table to allow an EMS to determine which alarms are currently active in the device. Two views of the active alarm table are supported by the agent:
  - acActiveAlarmTable in the proprietary AcAlarm MIB (this is a simple, one-row per alarm table that is easy to view with a MIB browser)
  - alarmActiveTable and alarmActiveVariableTable in the IETF standard AcAlarm MIB (rooted in the MIB tree)
- **Alarm History:** The device maintains a history of alarms that have been raised and traps that have been cleared to allow an EMS to recover any lost raised or cleared traps. Two views of the alarm history table are supported by the agent:
  - acAlarmHistoryTable in the proprietary AcAlarm MIB (this is a simple, one-row per alarm table that is easy to view with a MIB browser)
  - nlmLogTable and nlmLogVariableTable in the standard NOTIFICATION-LOG-MIB

5 Automatic Device Configuration

Large-scale deployment of devices calls for automated installation and setup capabilities. 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 devices may sometimes be pre-configured during the manufacturing process (commonly known as *private labeling*). Typically, a two-stage configuration process is implemented such that initial configuration includes only the basic configurations, while the final configuration is performed when the device is deployed in a live network.

Note: For support of automatic provisioning methods per device, see the table below.

Table 5-1: Automatic Provisioning Methods Support

Device	BootP / TFTP	DHCP 67	DHCP 66	Automatic Update Methods				SNMP (EMS)
				HTTP/S	TFTP	FTP	NFS	
MP-11x/124	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Mediant 600	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Mediant 1000	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Mediant 800 MSBG	Only in Rescue mode, on LAN side	No	No	Yes	Yes	Yes	Yes	Only VoIP configuration
Mediant 800 Gateway & SBC	Only in Rescue mode	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Mediant 1000 MSBG	Only in Rescue mode, on LAN side	No	No	Yes	Yes	Yes	Yes	Only VoIP configuration
Mediant 1000 Gateway & SBC	Only in Rescue mode	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Mediant 2000	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Mediant 3000	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Mediant 4000	Only in Rescue mode	No	No	Yes	Yes	Yes	Yes	Yes

5.1 Automatic Configuration Options

These available options for performing fast, automatic device configuration includes the following:

- Local configuration using BootP/TFTP server (see "Local Configuration Server with BootP/TFTP" on page 136)
- Configuration using DHCP (see "DHCP-based Configuration Server" on page 136)
- Configuration using DHCP Option 67 (see "Configuration using DHCP Option 67" on page 137)
- Configuration using DHCP Option 66 (see "TFTP Configuration using DHCP Option 66" on page 138)
- Configuration using HTTP (see "HTTP-based Automatic Updates" on page 138)
- Configuration using FTP or NFS ("Configuration using FTP or NFS" on page 139)
- Configuration using AudioCodes Element Management System (see "Configuration using AudioCodes EMS" on page 139)

5.1.1 Local Configuration Server with BootP/TFTP

Local configuration server with BootP/TFTP provides the most efficient and easiest method for automatic configuration, where configuration occurs at a staging warehouse, as described below:

1. A computer running BootP and TFTP software is located in a staging warehouse.
2. A standard *ini* configuration file is prepared and located in the TFTP directory.
3. BootP is configured with the MAC address of each device.
4. Each device is connected to the network and powered-up.
5. The BootP reply contains the cmp and *ini* file names entered in the 'Boot File' field. The device retrieves these files using BootP and stores them in its flash memory.
6. If auxiliary files are required (call progress tones etc.), they may be specified in the *ini* file and downloaded from the same TFTP server.
7. When the devices' LEDs turn green (i.e., files successfully loaded to the devices), the devices may be disconnected and shipped to the customer.
8. Local IP addressing at the customer site would typically be provided by DHCP.

For additional information, see "Using BootP / DHCP" on page 16.

5.1.2 DHCP-based Configuration Server

This alternative is similar to the setup described in "Local Configuration Server with BootP/TFTP" on page 136, except that DHCP is used instead of BootP. The DHCP server may be specially configured to automatically provide devices with a temporary IP address so that individual MAC addresses are not required. In this method, configuration occurs at a staging warehouse. For additional information, see "Using BootP / DHCP" on page 16.

Below is a sample configuration file for 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 "MP118_SIP_5.00A.001.cmp –fb;mp118.ini";
          option routers                  10.31.0.1;
          option subnet-mask              255.255.0.0;
    }
}
```

5.1.3 Configuration using DHCP Option 67

This method is suitable for deployments where DHCP server configuration is feasible at the customer site. Most DHCP servers allow configuring individual DHCP option values for different devices on the network. The DHCP configuration should be modified so that the device receives a configuration URL in option 67, along with IP addressing and DNS server information. The DHCP response is processed by the device upon startup, and consequently the HTTP server specified by the configuration URL is contacted to complete the configuration. This method does not require additional servers at the customer premises and is NAT-safe.

Below is an example of a Linux DHCP configuration file (dhcpd.conf) 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;
    }
}
```

5.1.4 TFTP Configuration using DHCP Option 66

This method is suitable when the customer's network contains a provisioning TFTP server for all network equipment, without being able to distinguish between AudioCodes and non-AudioCodes devices.

Upon startup, the device searches for Option 66 in the DHCP response. If Option 66 contains a valid IP address, the device attempts to download (through TFTP), a file named after the device's MAC address, e.g., "00908f0130aa.ini". This method requires a configuration server at the customer premises.

This method loads the configuration file to the device as a one-time action; the download is only repeated if the device is manually restored to factory defaults (by pressing the hardware reset button for 10 seconds while the Ethernet cable is not connected) and the DHCP is enabled (see note below). Note that access to the core network using TFTP is not NAT-safe.

Note: For TFTP configuration using DHCP Option 66, you must enable DHCP on your device:

- DHCPEnable = 1
- DHCPRequestTFTPParams = 1

When restoring the device to factory defaults, these parameters return to disabled.

5.1.5 HTTP-based Automatic Updates

An HTTP (or HTTPS) server can be placed in the customer's core network where configuration and software updates are available for download. This alternative does not require additional servers at the customer premises and is NAT-safe. For example, assume the core network HTTP server is https://www.corp.com. A master configuration *ini* file should be placed on the HTTP server, e.g., https://www.corp.com/gateways/master.ini. This *ini* file could point to additional *ini* files, auxiliary files (e.g., voice prompts and call progress tones), and software upgrades cmp files, all on the HTTP server or other HTTP servers in the core network.

The main advantage of this method is that the HTTP configuration can be checked periodically when the device is deployed at the customer site; HTTP(S) is not sensitive to NAT devices, allowing configuration whenever needed without on-site intervention.

For additional security, the URL may contain a different port, and a user name and password.

The devices should only be configured with the URL of the initial *ini* file. There are several methods for performing this:

- Using methods described in "DHCP-based Configuration Server" on page 136 or above, via TFTP at a staging warehouse. The *ini* file parameter controlling the configuration URL is IniFileURL.
- Private labeling.
- Using DHCP option 67 (see method described in "Configuration using DHCP Option 67" on page 137).
- Manually on-site, using the RS-232 port or Web interface.

When the device is deployed at the customer site, local DHCP provides IP addressing and DNS server information. The device can then contact the HTTP server at the core network and complete 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* becomes: http://corp.com/config-00908f030012.ini
- *http://corp.com/<IP>/config.ini* becomes: http://corp.com/192.168.0.7/config.ini

Software upgrades may be performed using the parameter CmpFileURL. Inclusion of this parameter in the master *ini* file causes the devices to download and store the specified software image.

For additional information, see "Automatic Update Mechanism" on page 21.

5.1.6 Configuration using FTP or NFS

Some networks block access to HTTP(S). The Automatic Update facility provides limited support for FTP/FTPS connectivity. However, periodic polling for updates is not possible (since these protocols don't support conditional fetching, i.e., updating files only if it is changed on the server).

The difference between this method and methods described in "HTTP-based Automatic Updates" on page 138 and "Configuration using DHCP Option 67" on page 137 is simply the protocol in the URL -- 'ftp' instead of 'http'. NFS v2/v3 is supported as well.

Note: Unlike FTP, NFS is not NAT-safe.

5.1.7 Configuration using AudioCodes EMS

AudioCodes EMS server functions as a core-network provisioning server. The device's SNMP Manager should be configured with the IP address of the EMS server, using one of the methods detailed in the previous sections. As soon as a registered device contacts the EMS server through SNMP, the EMS server 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.

5.2 Loading Files Securely (Disabling TFTP)

The TFTP protocol is not considered secure; some network operators block it using a firewall. It is possible to disable TFTP completely, using the *ini* file parameter EnableSecureStartup (set to 1). This way, secure protocols such as HTTPS may be used to fetch the device configuration.

➢ To download the ini file to the device using HTTPS instead of TFTP:

1. Prepare the device's configuration file on an HTTPS server, and obtain a URL to the file (e.g., https://192.168.100.53/gateways.ini).
2. Enable DHCP if necessary.
3. Enable SSH and connect to it (see "Starting a CLI Management Session" on page 25).
4. In the CLI, use the *ini* file parameters IniFileURL (for defining the URL of the configuration file) and EnableSecureStartup (for disabling TFTP), and then restart the device with the new configuration:

```
/conf/scp IniFileURL https://192.168.100.53/gateways.ini
/conf/scp EnableSecureStartup 1
/conf/sar bootp
```

Note: Once Secure Startup has been enabled, it can only be disabled by setting EnableSecureStartup to 0 using the CLI. Loading a new *ini* file using BootP/TFTP is not possible until EnableSecureStartup is disabled.

6 Security

This section describes the security mechanisms and protocols implemented on the device. The following list specifies the available security protocols and their objectives:

- IPSec and IKE protocols are part of the IETF standards for establishing a secured IP connection between two applications. IPSec and IKE are used in conjunction to provide security for control and management protocols but not for media (see "IPSec and IKE" on page 141).
- SSL (Secure Socket Layer) / TLS (Transport Layer Security). The SSL / TLS protocols are used to provide privacy and data integrity between two communicating applications over TCP/IP. They are used to secure the following applications: SIP Signaling (SIPS), Web access (HTTPS) and Telnet access (see "SSL/TLS" on page 148).
- Secured RTP (SRTP) according to RFC 3711 - used to encrypt RTP and RTCP transport (see "SRTP" on page 150).
- RADIUS (Remote Authentication Dial-In User Service) - RADIUS server is used to enable multiple-user management on a centralized platform (see "RADIUS Login Authentication" on page 152).
- Internal Firewall for filtering unwanted inbound traffic (see "Internal Firewall" on page 155).

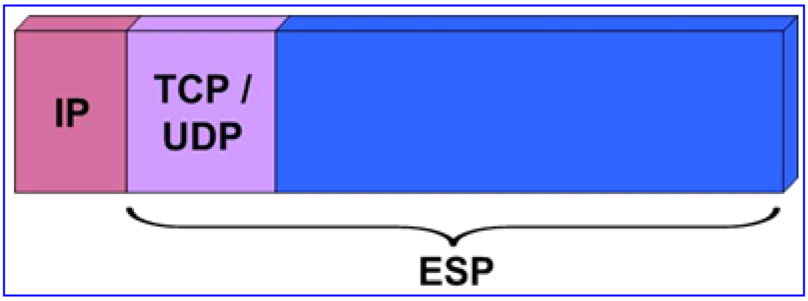
6.1 IPSec and IKE

Note: This section is not applicable to MSBG series, Mediant 800 Gateway & E-SBC, and Mediant 1000B Gateway & E-SBC.

IP security (IPSec) and Internet Key Exchange (IKE) protocols are part of the IETF standards for establishing a secured IP connection between two applications (also referred to as peers). Providing security services at the IP layer, IPSec and IKE are transparent to IP applications. IPSec and IKE are used together to provide security for control and management (e.g., SNMP and Web) protocols, but not for media (i.e., RTP, RTCP and T.38).

The IKE protocol is responsible for obtaining the IPSec encryption keys and encryption profile (known as IPSec Security Association or SA). IPSec is responsible for securing the IP traffic. This is accomplished by using the Encapsulation Security Payload (ESP) protocol to encrypt the IP payload (illustrated in the following figure).

Figure 6-1: IPSec Encryption



6.1.1 IKE (ISAKMP)

IKE is used to obtain the Security Associations (SA) between peers (the device and the application it's trying to contact). The SA contains the encryption keys and profile used by IPSec to encrypt the IP stream.

The IKE negotiation is separated into two phases: Main mode and Quick mode. The Main mode obtains a "master" encryption key (without any prior keys), and authenticates the peers to each other. Once initial security is set up, Quick mode sets up the encrypted IPSec tunnel.

The IKE negotiation is as follows:

- Main mode (creates a secured channel for the Quick mode):
  - **SA negotiation:** The peers negotiate their capabilities using up to four proposals. Each proposal includes three parameters: Encryption method, Authentication algorithm, and the DH group to use. The master key's lifetime is also negotiated in this stage. For detailed information on configuring proposals, see Proposal Configuration.
  - **Key exchange (DH):** The DH protocol is used to create the master key. DH requires both peers to agree on certain mathematical parameters, known as the "group".
  - **Authentication:** The two peers authenticate one another using a pre-shared key (configured in the IPSec association table) or by using certificate-based authentication. For information regarding authentication methods, see Peer Configuration.
- **Quick Mode (creates IPSec tunnels):**
  - **SA negotiation:** An IPSec SA is created by negotiating encryption and authentication capabilities, using the same proposal mechanism as in Main mode.
  - **Key exchange:** A symmetrical key is created for encrypting IPSec traffic; the peers communicate with each other in encrypted form, secured by the previously negotiated "master" key.

IKE Specifications:

- Authentication methods: pre-shared key or certificate-based authentication
- Main mode is supported for IKE Phase 1
- Diffie-Hellman group 1 or group 2
- Supported encryption algorithms: Data Encryption Standard (DES), Advanced Encryption Standard (AES), and 3DES
- Hash algorithms: SHA1 and MD5

6.1.2 IPSec

IPSec is responsible for encrypting and decrypting IP traffic.

The IPSec Security Association table defines up to 20 IP peers to which IPSec security is applied. IPSec can be applied to all traffic to and from a specific IP address. Alternatively, IPSec can be applied to a specific flow, specified by port (source or destination) and protocol type.

Each outgoing packet is analyzed and compared to the table. The packet's destination IP address (and optionally, destination port, source port and protocol type) are compared to each entry in the table. If a match is found, the device checks if an SA already exists for this entry. If it doesn’t, the IKE protocol is invoked (see "IKE (ISAKMP)" on page 142) and an IPSec SA is established. The packet is encrypted and transmitted. If a match is not found, the packet is transmitted without encryption.

Notes:

- An incoming packet whose parameters match one of the entries in the Association table, but is received without encryption is dropped.
- IPSec does not function properly if the device's IP address is changed on-the-fly. Therefore, reset the device after you change its IP address.

IPSec Specifications:

- Transport and Tunneling Mode
- Encapsulation Security Payload (ESP) only
- Supported encryption algorithms: AES, DES, and 3DES
- Supported hash types: SHA1 and MD5

6.1.3 Configuring IPSec and IKE

To enable IKE and IPSec processing, set the *ini* file parameter EnableIPSec to 1. Note that on some devices the channel capacity may be reduced even if no IPSec peers are configured.

6.1.3.1 Configuring Peers

Up to 20 peers (hosts or networks) can be defined for IPSec/IKE. These can be defined in the Web and ini file. Each of the entries in the IPSec Security Association table defines both Main and Quick modes for a single peer.

The IPSec/IKE parameters can also be configured using the *ini* file table parameter IPsecSATable. Each line in the table refers to a different IPSec/IKE peer. The following is an example of an IPSec/IKE table:

```
[ IPsecSATable ]
FORMAT IPsecSATable_Index = IPsecSATable_RemoteEndpointAddressOrName,
IPsecSATable_AuthenticationMethod, IPsecSATable_SharedKey,
IPsecSATable_SourcePort, IPsecSATable_DestPort, IPsecSATable_Protocol,
IPsecSATable_Phase1SaLifetimeInSec, IPsecSATable_Phase2SaLifetimeInSec,
IPsecSATable_Phase2SaLifetimeInKB, IPsecSATable_DPDmode,
IPsecSATable_IPsecMode, IPsecSATable_RemoteTunnelAddress,
IPsecSATable_RemoteSubnetIPAddress,
IPsecSATable_RemoteSubnetPrefixLength, IPsecSATable_InterfaceName;
IPsecSATable 1 = 0, 10.3.2.73, 0, 123456789, 0, 0, 0, 0, 28800, 3600, ;
[ \IPsecSATable ]
```

In the above example, a single IPSec/IKE peer (10.3.2.73) is configured. Pre-shared key authentication is selected, with the pre-shared key set to 123456789. In addition, a lifetime of 28800 seconds is selected for IKE and a lifetime of 3600 seconds is selected for IPSec.

6.1.3.2 Configuring Proposals

IKE can be configured with up to four proposal settings. Each proposal defines an encryption algorithm, an authentication algorithm, and a Diffie-Hellman group identifier. The same set of proposals apply to both Main and Quick mode.

Proposals are configured using the IPSec/IKE Proposal table (Web and *ini* file table parameter IPsecProposalTable).

For the *ini* file table parameter, each line defines a single proposal; the order of proposals offered by IKE is determined by the order of lines in the table. The following is an example of a proposal configuration table.

```
[ IPsecProposalTable ]
FORMAT IPsecProposalTable_Index = IPsecProposalTable_EncryptionAlgorithm,
IPsecProposalTable_AuthenticationAlgorithm, IPsecProposalTable_DHGroup;
IPsecProposalTable 0 = 3, 2, 1;
IPsecProposalTable 1 = 2, 2, 1;
[ \IPsecProposalTable ]
```

In the example above, two proposals are defined:

- Proposal 0: AES, SHA1, DH group 2
- Proposal 1: 3DES, SHA1, DH group 2

Note that with this configuration, neither DES nor MD5 can be negotiated.

6.1.3.3 IPSec Configuration Table Confidentiality

Since the pre-shared key parameter of the IKE table must remain undisclosed, measures are taken by the *ini* file, Web interface and SNMP agent to maintain this parameter's confidentiality. In the Web interface, a list of asterisks is displayed instead of the pre-shared key. In SNMP, the pre-shared key parameter is a write-only parameter and cannot be read. In the *ini* file, an asterisk is displayed instead of the pre-shared key.

6.1.4 Dead Peer Detection (RFC 3706)

When two peers communicate with IKE and IPSec, the situation may arise in which connectivity between the two goes down unexpectedly. In such cases, there is often no way for IKE and IPSec to identify the loss of peer connectivity. As such, the Security Associations (SA) remain active until their lifetimes naturally expire, resulting in a 'black hole' situation both peers discard all incoming network traffic.

This situation may be resolved by performing periodic message exchanges between the peers. When no reply is received, the sender assumes SA's are no longer valid on the remote peer and attempts to renegotiate.

The device can be configured to query the liveliness of its Internet Key Exchange (IKE) peer at regular intervals or on-demand, using the Dead Peer Detection (DPD) feature. DPD is automatically negotiated.

To activate the DPD feature, the *ini* file parameter DPDMode (in the IPSec Security Association table) must be set to one of the below values:

- [0] = Disabled (default).
- [1] = Periodic message exchanges at regular intervals.
- [2] = On-demand message exchanges as needed (i.e., before sending data to the peer). If the liveliness of the peer is questionable, the device sends a DPD message to

query the status of the peer. If the device has no traffic to send, it never sends a DPD message.

6.1.5 Certificate Revocation Checking

Some Public-Key Infrastructures (PKI) can revoke a certificate after it has been issued. The device, which employs SSL/TLS and IPSec, may be configured to check whether a peer's certificate has been revoked, using the Online Certificate Status Protocol (OCSP).

To configure OCSP, the following *ini* file parameters must be configured:

- OCSPEnable
- OCSPServerIP (and optionally OCSPSecondaryServerIP)
- OCSPServerPort
- OCSPDefaultResponse

For a description of these parameters, refer to the device's *User's Manual*.

When OCSP is enabled, the device queries the OCSP server for revocation information whenever a peer certificate is received (IPSec, TLS client mode, or TLS server mode with mutual authentication). Note that a second OCSP server for redundancy can be configured (OCSPSecondaryServerIP).

Notes:

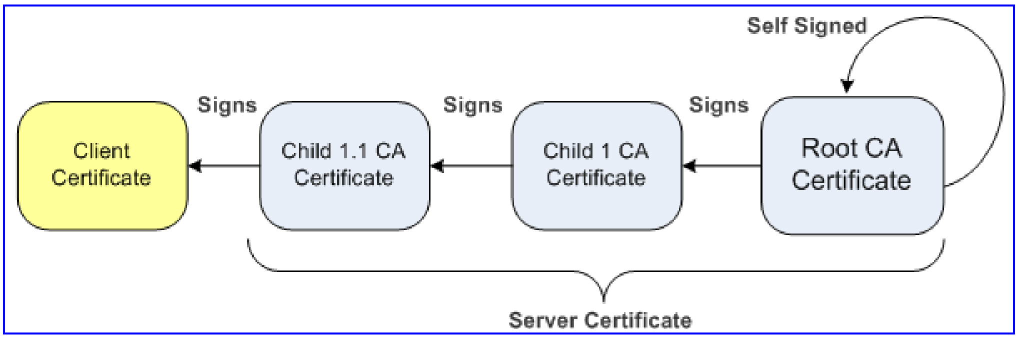
- The device does not query OCSP for its own certificate.
- Some PKIs do not support OCSP, but generate Certificate Revocation Lists (CRLs). In such a scenario, set up an OCSP server such as OCSPD.

6.1.6 Certificate Chain

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.

In order for the device to trust a whole chain of certificates, you need to combine the certificates into one text file (using a text editor). Upload the file as the "Trusted Root Certificate Store" file as discussed above.

Figure 6-2: Certificate Chain Hierarchy



Note: When configuring a trusted chain of certificates, the file size is limited to up to 9000 characters (including the certificate's headers).

6.2 Secure Shell

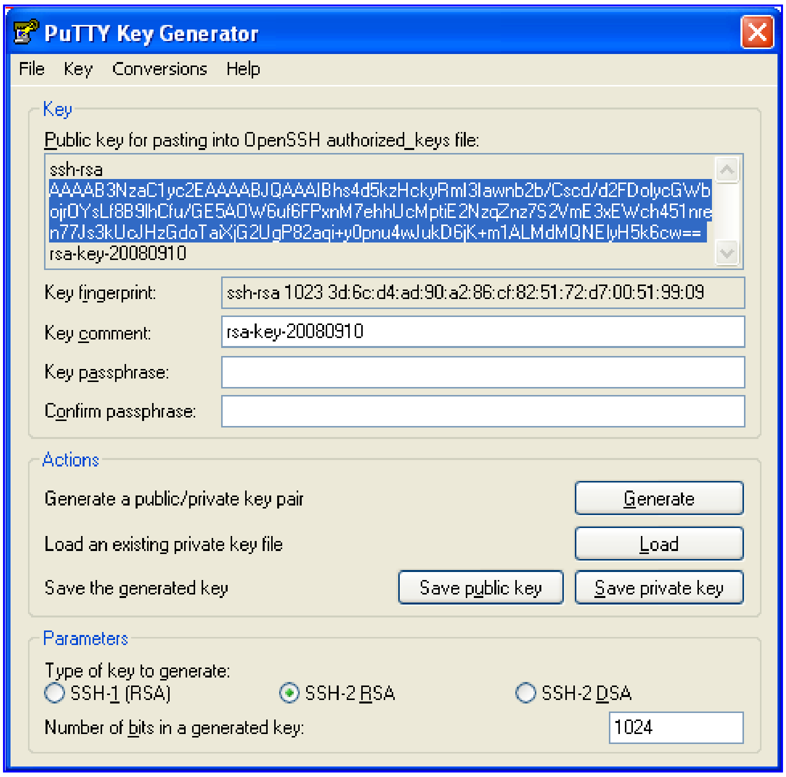
The device's command-line interface (CLI) may be accessed using Telnet. However, unless configured for TLS mode, Telnet is not secure as it requires passwords to be transmitted in clear text. To overcome this, Secure SHell (SSH) is used, which is the de-facto standard for secure CLI. SSH 2.0 is a protocol 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, which can be downloaded from http://www.chiark.greenend.org.uk/~sgtatham/putty/.

By default, SSH uses the same user name and password as the Telnet server and Web server. In addition, SSH supports 1024/2048-bit RSA public keys, which provide carrier-grade security. Follow the instructions below to configure the device with an administrator RSA key as a means of strong authentication.

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

1. Run the file *puttygen.exe*; the PuTTY Key Generator program starts, displaying the main window.
2. Under the 'Parameters' group, perform the following:
   a. Select the option 'SSH-2 RSA'.
   b. In the field 'Number of bits in a generated key', enter "1024" bits.
3. Under the 'Actions' group, click **Generate**, and then follow the on-screen instructions.
4. Under the 'Actions' group, save the new private key to a file (*.ppk) on your PC, by clicking **Save private key**.
5. Under the 'Key' group, select the displayed encoded text between "ssh-rsa" and "rsa-key-….", as shown in the example below:



Public key for pasting into OpenSSH authorized_keys file:

```
ssh-rsa
AAAAB3NzaC1yc2EAAAABJQAAAIBhs4d5kzHckyRml3lawnb2b/Cscd/d2FDolycGWb
ojrOYsLf8B9lhCfu/GE5AOW6uf6FPxnM7ehhUcMptiE2NzqZnz7S2VmE3xEWch451nre
n77Js3kUcJHzGdoTaiXjG2UgP82aqi+y0pnu4wJukD6jK+m1ALMdMQNElyH5k6cw==
rsa-key-20080910
```

Key fingerprint: ssh-rsa 1023 3d:6c:d4:ad:90:a2:86:cf:82:51:72:d7:00:51:99:09

Key comment: rsa-key-20080910

Key passphrase:

Confirm passphrase:

Generate a public/private key pair [Generate]

Load an existing private key file [Load]

Save the generated key [Save public key] [Save private key]

Type of key to generate:
○ SSH-1 (RSA) ● SSH-2 RSA ○ SSH-2 DSA

Number of bits in a generated key: 1024

6. Open the device's ini file, and then paste the public key (that you copied in Step 5) as the value for the parameter SSHAdminKey, as shown below:

```
SSHAdminKey = AAAAB3NzaC1yc2EAAAABJQ…
```

7. Load the ini file to the device.
8. Run the file *PuTTY.exe*; the PuTTY Configuration program starts.
9. In the 'Category' tree, drill down the tree by selecting **Connection,** then **SSH**, and then **Auth**; the 'Options controlling SSH authentication' pane appears.
10. Under the 'Authentication parameters' group, click **Browse** and then locate the private key file that you created and saved in Step 4.
11. Connect to the device with SSH, using the user name "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:

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

   A new key is created in the file *admin.key* and the public portion is saved to the file *admin.key.pub*.
2. Open the file *admin.key.pub*, and then copy the long encoded string from "ssh-rsa" up to the white-space.

3. Open the device’s ini file, and then set the SSHAdminKey to the value copied in Step 2, e.g.:

```
SSHAdminKey = AAAAB3NzaC1yc2EAAAABJQ…
```

4. Load the ini file to the device.
5. Connect to the device with SSH, using the following command:

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

   where *xx.xx.xx.xx* is the device's IP address.

   RSA key negotiation occurs automatically and no password is required.

For additional security, you can set the *ini* file parameter SSHRequirePublicKey to 1. This ensures that SSH access is only possible by using the RSA key and not by using user name and password.

6.3 SSL / TLS

Secure Socket Layer (SSL), also known as Transport Layer Security (TLS) is the method used to secure the device's SIP signaling connections, Web interface, and Telnet server. The SSL protocol provides confidentiality, integrity, and authenticity between two communicating applications over TCP/IP.

Specifications for the SSL/TLS implementation include the following:

- **Transports:** SSL 2.0, SSL 3.0, TLS 1.0.
- **Ciphers:** DES, RC4 compatible, Advanced Encryption Standard (AES).
- **Authentication:** X.509 certificates (CRLs are currently not supported). The device supports the receipt of wildcards (‘*’) in X.509 Certificates when establishing TLS connections. These wildcards can be part of the CN attribute of the Subject field or the DNSName attribute of the SubjectAltName field.

> **Tip:** A common security practice is to disable SSLv2/SSLv3 and use only TLSv1. This can be achieved by setting the *ini* file parameter TLSVersion to 1. If using Microsoft Internet Explorer, ensure you disable SSL 2.0 / SSL 3.0 and enable TLS 1.0 in Internet Explorer (**Tools** > **Internet Options** > **Advanced**).

6.3.1 SIP Over TLS (SIPS)

The device uses TLS over TCP to encrypt SIP transport and (optionally) to authenticate it. To enable TLS on the device, set the selected transport type to TLS (SIPTransportType = 2). In this mode, the device initiates a TLS connection only for the next network hop. To enable TLS all the way to the destination (over multiple hops), set EnableSIPS to 1. When a TLS connection with the device is initiated, the device also responds using TLS, regardless of the configured SIP transport type (in this case, the parameter EnableSIPS is also ignored).

TLS and SIPS use the Certificate Exchange process described in Server Certificate Replacement and Client Certificates in the *User's Manual*. To change the port number used for SIPS transport (by default, 5061), use the parameter TLSLocalSIPPort.

When SIPS is implemented, it is sometimes required to use two-way authentication. When acting as the TLS server (in a specific connection), it is possible to demand the authentication of the client’s certificate. To enable two-way authentication on the device, set the *ini* file parameter SIPSRequireClientCertificate to 1. For information on installing a client certificate, refer to Client Certificates described in the *User's Manual*.

6.3.2 Secured HTTPS Web Interface Configuration

For additional security, you can configure the Web interface to accept only secured (HTTPS) connections by setting the parameter HTTPSOnly to 1 (described in the device's *User's Manual*). You can also change the port number used for the secured Web server (by default, 443), by changing the *ini* file parameter, HTTPSPort (described in the device's *User's Manual*).

➢ **To use the secured Web interface:**

1. Access the device using the following URL: https://[host name or IP address]

   Depending on the browser's configuration, a security warning dialog may be displayed. The reason for the warning is that the device initial certificate is not trusted by your PC. The browser may allow you to install the certificate, thus skipping the warning dialog the next time you connect to the device.

2. If you are using Internet Explorer, click **View Certificate**, and then **Install Certificate**.

3. The browser also warns you if the host name used in the URL is not identical to the one listed in the certificate. To solve 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:

```
# This is a sample HOSTS file used by Microsoft TCP/IP for
Windows.

# Location: C:\WINDOWS\SYSTEM32\DRIVERS\ETC\hosts

#

127.0.0.1    localhost
10.31.4.47   ACL_280152
```

6.3.3 Secured Telnet

To enable the embedded Telnet server on the device, set the parameter TelnetServerEnable (described in 'Web and Telnet Parameters' in the device's *User's Manual*) to 1 (standard mode) or 2 (SSL mode); no information is transmitted in the clear when SSL mode is used.

If the Telnet server is set to SSL mode, a special Telnet client is required on your PC to connect to the Telnet interface over a secured connection; examples include C-Kermit for UNIX, Kermit-95 for Windows, and AudioCodes' acSSLTelnet utility for Windows (that requires prior installation of the free OpenSSL toolkit). Contact AudioCodes to obtain the acSSLTelnet utility.

For security reasons, some organizations require the display of a proprietary notice upon starting a Telnet session. The following is an example of a configuration *ini* file for defining such a message:

```
[ WelcomeMessage ]
FORMAT WelcomeMessage_Index = WelcomeMessage_Text ;
WelcomeMessage 01 = "WARNING! This computer system and network is
PRIVATE and PROPRIETARY and may" ;
WelcomeMessage 02 = "only be accessed by authorized users.
Unauthorized use of this computer" ;
WelcomeMessage 03 = "system or network is strictly prohibited and
may be subject to criminal" ;
WelcomeMessage 04 = "prosecution, employee discipline up to and
including discharge, or the" ;
WelcomeMessage 05 = "termination of vendor/service contracts. The
owner, or its agents, may" ;
WelcomeMessage 06 = "monitor any activity or communication on the
computer system or network." ;
WelcomeMessage 07 = "The owner, or its agents, may retrieve any
information stored within the" ;
WelcomeMessage 08 = "computer system or network. By accessing and
using this computer system or" ;
WelcomeMessage 09 = "network, you are consenting to such
monitoring and information retrieval for" ;
WelcomeMessage 10 = "law enforcement and other purposes. Users
should have no expectation of" ;
WelcomeMessage 11 = "privacy as to any communication on or
information stored within the computer" ;
WelcomeMessage 12 = "system or network, including information
stored locally or remotely on a hard" ;
WelcomeMessage 13 = "drive or other media in use with this
computer system or network." ;
[ /WelcomeMessage ]
```

6.4 SRTP

The device supports Secured RTP (SRTP) according to RFC 3711. SRTP is used to encrypt RTP and RTCP transport as it is best suited for protecting VoIP traffic.

SRTP requires a Key Exchange mechanism that is performed according to RFC 4568 – “Session Description Protocol (SDP) Security Descriptions for Media Streams”. The Key Exchange is executed by adding 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 to use. If negotiation of the encryption data is successful, the call is established.

SRTP implementation supports the following suites:

- AES_CM_128_HMAC_SHA1_32
- AES_CM_128_HMAC_SHA1_80
- ARIA_128_BIT
- ARIA_192_BIT

All other suites are ignored.

The only supported key parameter is Master Key Identifier (MKI) value. When the device is the offering side, it generates an MKI of a size defined by the *ini* file parameter SRTPTxPacketMKISize. The length of the MKI is limited to four bytes. If the remote side sends a longer MKI, then the key is ignored. The key lifetime field is not supported. However, if it is included in the key it is ignored and the call does not fail.

The device supports the following Session parameters (as defined in RFC 4568, SDP Security Descriptions for Media Streams):

- UNENCRYPTED_SRTP
- UNENCRYPTED_SRTCP
- UNAUTHENTICATED_SRTP

Session parameters should be the same for the local side and remote side. When the device is the offering side, the session parameters are defined according to the following *ini* file parameters: RTPEncryptionDisableTx, RTCPEncryptionDisableTx, and RTPAuthenticationDisableTx. 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
```

Use the parameter MediaSecurityBehaviour (described in the device's *User's Manual*) to select the device's mode of operation that determines the behavior of the device if negotiation of the cipher suite fails:

- Mandatory: the call is terminated. Incoming calls that don’t include encryption information are rejected.
- Preferable: an unencrypted call is established. Incoming calls that don’t include encryption information are accepted.

To enable SRTP, set the parameter EnableMediaSecurity to 1 (described in the device's *User's Manual*).

Notes:

- SRTP support is according to the selected DSP version template.
- When SRTP is used, the channel capacity is reduced (refer to the parameter EnableMediaSecurity).

6.5 RADIUS Login Authentication

Users can enhance the security and capabilities of logging to the device’s Web and Telnet embedded servers by using a Remote Authentication Dial-In User Service (RADIUS) to store numerous user names, passwords and access level attributes (Web only), allowing multiple user management on a centralized platform. RADIUS (RFC 2865) is a standard authentication protocol that defines a method for contacting a predefined server and verifying a given name and password pair against a remote database in a secure manner.

When accessing the Web and Telnet servers, users must provide a valid user name and password. When RADIUS authentication isn’t used, the user name and password are authenticated with the Web interface's user names and passwords of the primary or secondary accounts (refer to 'User Accounts' in the device's *User's Manual*) or with the Telnet server’s user name and password stored internally in the device's memory. When RADIUS authentication is used, the device doesn’t store the user name and password but simply forwards them to the pre-configured RADIUS server for authentication (acceptance or rejection). The internal Web/Telnet passwords can be used as a fallback mechanism in case the RADIUS server doesn’t respond (configured by the parameter BehaviorUponRadiusTimeout). Note that when RADIUS authentication is performed, the Web/Telnet servers are blocked until a response is received (with a timeout of 5 seconds).

RADIUS authentication requires HTTP basic authentication, meaning the user name and password are transmitted in clear text over the network. Therefore, users are recommended to set the parameter HttpsOnly to 1, to force the use of HTTPS, since the transport is encrypted.

6.5.1 Setting Up a RADIUS Server

The following examples refer to FreeRADIUS, a free RADIUS server that can be downloaded from www.freeradius.org. Follow the directions on that site for information on installing and configuring the server. If you use a RADIUS server from a different vendor, refer to its appropriate documentation.

➢ To set up a RADIUS server

1. Define the device as an authorized client of the RADIUS server, with a predefined ‘shared secret’ (a password used to secure communication) and a vendor ID. Below is an example of the file clients.conf (FreeRADIUS client configuration).

```
#
# clients.conf - client configuration directives
#
client 10.31.4.47 {
        secret          = FutureRADIUS
        shortname       = tp1610_master_tpm
}
```

2. If access levels are required, set up a VSA dictionary for the RADIUS server and select an attribute ID that represents each user's access level. The following example shows a dictionary file for FreeRADIUS (FreeRADIUS Client Configuration) that defines the attribute ‘ACL-Auth-Level’ with ID=35.

```
#
# 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. In the RADIUS server, define the list of users authorized to use the device, using one of the password authentication methods supported by the server implementation. The following example 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

larry   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’, vendor ID and VSA access level identifier (if access levels are used) used by the RADIUS server.

5. Configure the device's relevant parameters according to "Configuring RADIUS Support" on page 153.

6.5.2 Configuring RADIUS Support

The procedure below describes how to configure RADIUS for the device using the Web interface. For information on the RADIUS parameters, refer to the device's *User's Manual*.

➢ To configure RADIUS support using the Web interface:

1. Access the Web interface (refer to the device's *User's Manual*).

2. Open the ‘RADIUS Settings’ screen (**Configuration** tab > **System** menu > **Management** > **RADIUS Settings**).

3. Under section ‘General RADIUS Settings’, in the field ‘Enable RADIUS Access Control’, select ‘Enable’; the RADIUS application is enabled.

4. In the field ‘Use RADIUS for Web/Telnet Login’, select ‘Enable’; RADIUS authentication is enabled for Web and Telnet login.

5. Enter the RADIUS server IP address, port number and shared secret in the relevant fields.

6. Under section ‘General RADIUS Authentication’, in the field ‘Device Behavior Upon RADIUS Timeout’, select the device’s operation if a response isn’t received from the RADIUS server after the 5 seconds timeout expires:

   - Deny Access: the device denies access to the Web and Telnet embedded servers.
   - Verify Access Locally: the device checks the local user name and password.

7. In the field ‘Local RADIUS Password Cache Timeout’, enter a time (in seconds); when this time expires, the user name and password verified by the RADIUS server becomes invalid and a user name and password must be re-validated with the RADIUS server.

8. In the field ‘Local RADIUS Password Cache Mode’, select the device's mode of operation regarding the above-mentioned ‘Local RADIUS Password Cache Timer’ option:

   - Reset Timer Upon Access: upon each access to a Web screen, the timer resets (reverts to the initial value configured in the previous step).
   - Absolute Expiry Timer: when you access a Web screen, the timer doesn’t reset but rather continues decreasing.

9. In the field ‘RADIUS VSA Vendor ID’, enter the vendor ID you configured in the RADIUS server:

10. When using the Web access-level mechanism, perform one of the following options:

    - When RADIUS responses include the access level attribute:
      In the field ‘RADIUS VSA Access Level Attribute’, enter the code that indicates the access level attribute in the Vendor Specific Attributes (VSA) section of the received RADIUS packet.
    - When RADIUS responses don’t include the access level attribute:
      In the field ‘Default Access Level’, enter the default access level that is applied to all users authenticated by the RADIUS server.

11. In the field ‘Secured Web Connection (HTTPS)’, select ‘HTTPS only’. It is important you use HTTPS (secure Web server) when connecting to the device over an open network, since the password is transmitted in clear text. Similarly, for Telnet, use SSL TelnetServerEnable = 2 or SSH (see "Secured Telnet" on page 149).

12. Save the changes so they are available after a power fail.

13. Reset the device (refer to the device's *User's Manual*).

After reset, when accessing the Web or Telnet servers, use the user name and password you configured in the RADIUS database. The local system password is still active and can be used when the RADIUS server is down.

➢ To configure RADIUS support on the device using the *ini* file:

1. Add the following parameters to the *ini* file.

   - EnableRADIUS = 1
   - WebRADIUSLogin = 1
   - RADIUSAuthServerIP = IP address of RADIUS server
   - RADIUSAuthPort = port number of RADIUS server, usually 1812
   - SharedSecret = your shared secret
   - HTTPSOnly = 1

- RadiusLocalCacheMode = 1
- RadiusLocalCacheTimeout = 300
- RadiusVSAVendorID = your vendor's ID
- RadiusVSAAccessAttribute = code that indicates the access level attribute
- DefaultAccessLevel = default access level (0 to 200)

2. Authenticating via RADIUS with credentials in the URL:
   - The device is capable of authenticating via RADIUS server when the UserName/Password are in the URL, e.g.,:

     http://10.4.4.112/Forms/RadiusAuthentication?WSBackUserName=Guyy&WSBackPassword=1234
   - This method is applicable when using RADIUS server with HTTP basic authentication.
3. To set this feature, use RADIUS with Basic authentication settings:
   a. Default settings: You are prompted for your login every time you connect to the blade.
   b. Enable RADIUS configuration as described above.
   c. Enable Basic HTTP authentication settings.
   d. Connect to the device using a URL as in the example.

This feature is restricted to five simultaneous users only.

6.6 Internal Firewall

The device accommodates an internal access list facility, allowing the security administrator to define VoIP network traffic filtering rules. The access list provides the following features:

- Block traffic from known malicious sources
- Only allow traffic from known friendly sources, and block all others
- Mix allowed and blocked network sources
- Limit traffic to a predefined rate (blocking the excess)
- Limit traffic to specific protocols, and specific port ranges on the device

For each packet received on the network interface, the table is scanned from the top until a matching rule is found (or the table end is reached). This rule can either block the packet or allow it; however it is important to note that subsequent rules aren't scanned. If the table end is reached without a match, the packet is accepted.

Each rule is composed of the following fields (described in the device's *User's Manual*):

- IP address (or DNS name) of source network
- IP network mask (prefix length)
- Destination UDP/TCP ports (on this device)
- Protocol type
- Maximum packet size, byte rate per second, and allowed data burst
- Action upon match (allow or block)

The internal firewall can be configured using the *ini* file or the Web interface (refer to the device's *User's Manual*).

Below is an example of an access list definition via *ini* file (this can also be configured using the Web interface):

```
[ ACCESSLIST ]
FORMAT ACCESSLIST_Index = ACCESSLIST_Source_IP,
ACCESSLIST_Net_MaskPrefixLen, ACCESSLIST_Start_Port,
ACCESSLIST_End_Port, ACCESSLIST_Protocol, ACCESSLIST_Packet_Size,
ACCESSLIST_Byte_Rate, ACCESSLIST_Byte_Burst,
ACCESSLIST_Allow_Type;
ACCESSLIST 10 = mgmt.customer.com, 255.255.255.25532, 0, 80, tcp,
0, 0, 0, allow ;
ACCESSLIST 15 = 192.0.0.0, 255.0.0.08, 0, 65535, any, 0, 40000,
50000, block ;
ACCESSLIST 20 = 10.31.4.0, 255.255.255.024, 4000, 9000, any, 0, 0,
0, block; ACCESSLIST 22 = 10.4.0.0, 255.255.0.016, 4000, 9000,
any, 0, 0, 0, block ;
[\ACCESSLIST]
```

Explanation of the example access list:

- Rule #10: traffic from the host ‘mgmt.customer.com’ destined to TCP ports 0 to 80, is always allowed.
- Rule #15: traffic from the 192.xxx.yyy.zzz subnet, is limited to a rate of 40 Kbytes per second (with an allowed burst of 50 Kbytes). Note that the rate is specified in bytes, not bits, per second; a rate of 40000 bytes per second, nominally corresponds to 320 kbps.
- Rule #20: traffic from the subnet 10.31.4.xxx destined to ports 4000 to 9000 is always blocked, regardless of protocol.
- Rule #22: traffic from the subnet 10.4.xxx.yyy destined to ports 4000 to 9000 is always blocked, regardless of protocol.
- All other traffic is allowed.

More complex rules may be defined, relying on the ‘single-match’ process described above. Below is an advanced example of an access list definition via *ini* file:

```
[ ACCESSLIST ]
FORMAT ACCESSLIST_Index = ACCESSLIST_Source_IP,
ACCESSLIST_Net_MaskPrefixLen, ACCESSLIST_Start_Port,
ACCESSLIST_End_Port, ACCESSLIST_Protocol, ACCESSLIST_Packet_Size,
ACCESSLIST_Byte_Rate, ACCESSLIST_Byte_Burst,
ACCESSLIST_Allow_Type;
ACCESSLIST 10 = 10.0.0.0, 255.0.0.08, 0, 65535, any, 0, 40000,
50000, allow ;
ACCESSLIST 15 = 10.31.4.0, 255.255.255.024, 4000, 9000, any, 0, 0,
0, allow ;
ACCESSLIST 20 = 0.0.0.0, 0.0.0.00, 0, 65535, any, 0, 0, 0, block;
[\ACCESSLIST]
```

This access list (in the example above) consists of three rules:

- Rule #10: traffic from the subnet 10.xxx.yyy.zzz is allowed if the traffic rate does not exceed 40 KB/s.
- Rule #15: if a packet didn't match rule #10, that is, the excess traffic is over 40 KB/s, and coming from the subnet 10.31.4.xxx to ports 4000 to 9000, then it is allowed.
- Rule #20: all other traffic (which didn't match the previous rules), is blocked.

6.7 Network Security with IEEE 802.1X

Note: The following section is applicable only to MediaPack.

The device can operate as an IEEE 802.1X supplicant. IEEE 802.1X is a standard for port-level security on secure Ethernet switches; when a device is connected to a secure port, no traffic is allowed until the identity of the device is authenticated.

A typical 802.1X deployment consists of an Authenticator (secure LAN switch), an Access Server (e.g. RADIUS), and one or more supplicants. The Authenticator blocks all traffic on the secure port by default, and communicates with the supplicant via EAP-over-LAN frames. The supplicant provides credentials which are transmitted to the Access Server; if the Access Server determines that the credentials are valid, it instructs the Authenticator to authorize traffic on the secure port.

The device supports the following Extensible Authentication Protocol (EAP) variants:

- **MD5-Challenge (EAP-MD5):** authentication is performed using a user name and password, provisioned to the device using the parameters 802.1xUsername and 802.1xPassword.
- **Protected EAP (PEAPv0 with EAP-MSCHAPv2):** authentication uses 802.1xUsername and 802.1xPassword, however, the protocol used is MSCHAPv2 over an encrypted TLS tunnel

- **EAP-TLS:** the device's certificate is used to establish a mutually-authenticated TLS session with the Access Server. This requires prior configuration of the server certificate and root CA (see previous sections about certificate configuration). The parameter 802.1xUsername is used to identify the device, however, 802.1xPassword is ignored.

Note: The EAP variant is selected using the parameter 802.1xMode.

6.8 Network Port Usage

The following table lists the default TCP/UDP network port numbers used by the device. Where relevant, the table lists the *ini* file parameters that control the port usage and provide source IP address filtering capabilities.

Table 6-1: Default TCP/UDP Network Port Numbers

Port Number	Peer Port	Application	Notes
2	2	Debugging interface	Always ignored
22	-	SSH	Disabled by default (SSHServerEnable). Configurable (SSHServerPort), access controlled by WebAccessList. **Note:** Not applicable to MSBG, Mediant 800 Gateway & E-SBC, andMediant 1000 Gateway & E-SBC.
23	-	Telnet	Disabled by default (TelnetServerEnable). Configurable (TelnetServerPort), access controlled by WebAccessList
68	67	DHCP	Active only if DHCPEnable = 1 **Note:** Not applicable to MSBG, Mediant 800 Gateway & E-SBC, andMediant 1000 Gateway & E-SBC.
80	-	Web server (HTTP)	Configurable (HTTPPort), can be disabled (DisableWebTask or HTTPSOnly). Access controlled by WebAccessList
161	-	SNMP GET/SET	Configurable (SNMPPort), can be disabled (DisableSNMP). Access controlled by SNMPTrustedMGR
443	-	Web server (HTTPS)	Configurable (HTTPSPort), can be disabled (DisableWebTask). Access controlled by WebAccessList
500	-	IPSec IKE	Can be disabled (EnableIPSec) **Note:** Not applicable to MSBG series, Mediant 800 Gateway & E-SBC, andMediant 1000 Gateway & E-SBC.
4000, 4010 and up	-	RTP traffic	Base port number configurable (BaseUDPPort), fixed increments of 10. The number of ports used depends on the

Port Number	Peer Port	Application	Notes
			channel capacity of the device.
4001, 4011 and up	-	RTCP traffic	Always adjacent to the RTP port number
4002, 4012 and up	-	T.38 traffic	Always adjacent to the RTCP port number
5060	5060	SIP	Configurable (LocalSIPPort [UDP], TCPLocalSIPPort [TCP]).
5061	5061	SIP over TLS (SIPS)	Configurable (TLSLocalSIPPort)
(random) > 60000	514	Syslog	Configurable (SyslogServerPort). Disabled by default (EnableSyslog).
(random) > 60000	162	SNMP Traps	Can be disabled (DisableSNMP)
(random) > 60000	-	DNS client	

6.9 Recommended Practices

To improve network security, the following guidelines are recommended when configuring the device:

- Define the password of the primary Web user account (see 'Configuring the Web User Accounts' in the device's *User's Manual*) to a unique, hard-to-hack string. Do not use the same password for several devices as a single compromise may lead to others. Keep this password safe at all times and change it frequently.

- If possible, use a RADIUS server for authentication. RADIUS allows you to set different passwords for different users of the device, with centralized management of the password database. Both Web and Telnet interfaces support RADIUS authentication (see "RADIUS Login Authentication" on page 152).
(**Note:** RADIUS is not applicable to the 3000 Series.)

- If the number of users that access the Web and Telnet interfaces is limited, you can use the ‘Web and Telnet Access List’ to define up to ten IP addresses that are permitted to access these interfaces. Access from an undefined IP address is denied (see 'Configuring the Web and Telnet Access List' in the device's *User's Manual*).

- Use IPSec to secure traffic to all management and control hosts. Since IPSec encrypts all traffic, hackers cannot capture sensitive data transmitted on the network, and malicious intrusions are severely limited.

- Use HTTPS when accessing the Web interface. Set HTTPSOnly to 1 to allow only HTTPS traffic (and block port 80). If you don't need the Web interface, disable the Web server using the DisableWebTask parameter (Note that this is not applicable to 3000 Series).

- If you use Telnet, do not use the default port (23). Use SSL mode to protect Telnet traffic from network sniffing.

- If you use SNMP, do not leave the community strings at their default values as they can be easily guessed by hackers (see "SNMP Community Names" on page 70).

- Use a firewall to protect your VoIP network from external attacks. Network robustness may be compromised if the network is exposed to Denial of Service (DoS) attacks. DoS attacks are mitigated by Stateful firewalls. Do not allow unauthorized traffic to reach the device.

6.10 Legal Notice

By default, the device supports export-grade (40-bit and 56-bit) encryption due to US government restrictions on the export of security technologies. To enable 128-bit and 256-bit encryption on your device, contact your AudioCodes representative.

This product includes software developed by the OpenSSL Project for use in the OpenSSL Toolkit (http://www.openssl.org/).

This product includes cryptographic software written by Eric Young (eay@cryptsoft.com).

7 RTP Control Protocol Extended Reports (RTCP XR)

Notes:

- This section is applicable to all devices except MP.
- To support RTCP XR, the device must be installed with the RTCP XR Feature Key.

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 diagnosing problems. RTCP XR (RFC 3611) extends the RTCP reports defined in RFC 3550 by providing additional VoIP metrics.

RTCP XR information publishing is implemented in the device according to <draft-johnston-sipping-rtcp-summary-07>. This draft defines how a SIP User Agent (UA) publishes the detailed information to a defined collector.

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 using SNMP. The device can send RTCP XR reports to an Event State Compositor (ESC) server using PUBLISH messages. These reports can be sent at the end of each call (configured using RTCPXRReportMode) and according to a user-defined interval (RTCPInterval or DisableRTCPRandomize) between consecutive reports.

To enable RTCP XR reporting, the VQMonEnable *ini* file parameter must be set to 1. In addition, the device must be installed with the appropriate Software Upgrade Key. For a detailed description of the RTCP XR *ini* file parameters, refer to the device's *User's Manual*.

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 the metrics listed in the table below.

Table 7-1: RTCP XR Published VoIP Metrics

	Metric Name
General	Start Timestamp
	Stop Timestamp
	Call-ID
	Local Address (IP, Port & SSRC)
	Remote Address (IP, Port & SSRC)
Session Description	Payload Type
	Payload Description
	Sample Rate
	Frame Duration
	Frame Octets
	Frames per Packets
	Packet Loss Concealment

	Metric Name
	Silence Suppression State
Jitter Buffer	Jitter Buffer Adaptive
	Jitter Buffer Rate
	Jitter Buffer Nominal
	Jitter Buffer Max
	Jitter Buffer Abs Max
Packet Loss	Network Packet Loss Rate
	Jitter Buffer Discard Rate
Burst Gap Loss	Burst Loss Density
	Burst Duration
	Gap Loss Density
	Gap Duration
	Minimum Gap Threshold
Delay	Round Trip Delay
	End System Delay
	One Way Delay
	Interarrival Jitter
	Min Absolute Jitter
	Signal
	Signal Level
	Noise Level
	Residual Echo Return Noise
Quality Estimates	Listening Quality R
	RLQ Est. Algorithm
	Conversational Quality R
	RCQ Est. Algorithm
	External R In
	Ext. R In Est. Algorithm
	External R Out
	Ext. R Out Est. Algorithm
	MOS-LQ
	MOS-LQ Est. Algorithm
	MOS-CQ
	MOS-CQ Est. Algorithm
	QoE Est. Algorithm

8 RTP / RTCP Payload Types and Port Allocation

RTP Payload Types are defined in RFC 3550 and RFC 3551. We have added new payload types to enable advanced use of other coder types. These types are reportedly not used by other applications.

8.1 Payload Types Defined in RFC 3551

Table 8-1: Packet Types Defined in RFC 3551

Payload Type	Description	Basic Packet Rate (msec)
0	G.711 µ-Law	10, 20
2	G.726-32	10, 20
3	GSM-FR **Note:** Applicable only to 2000 Series and 3000 Series.	20
4	G.723 (6.3/5.3 kbps)	30
8	G.711 A-Law	10, 20
9	G.722	20
12	QCELP **Note:** Applicable only to 2000 Series.	20
18	G.729A/B	20
200	RTCP Sender Report	Randomly, approximately every 5 seconds (when packets are sent by channel)
201	RTCP Receiver Report	Randomly, approximately every 5 seconds (when channel is only receiving)
202	RTCP SDES packet	
203	RTCP BYE packet	
204	RTCP APP packet	

8.2 Defined Payload Types

The defined payload types are listed in the table below.

Note: Not all coders are supported on all devices. For a detailed description of supported coders, please refer to the device's *Release Notes*.

Table 8-2: Defined Payload Types

Payload Type	Description	Basic Packet Rate (msec)
3	MS-GSM	40
3	GSM-EFR	20
22	G.726-24	20
23	G.726-16	20
38	G.726-40	20
56	Transparent PCM	20
60	EVRC	20
64	AMR	20
64	AMR-WB	20
65	iLBC	20, 30
68	EVRC-B (4GV)	20
96	DTMF relay per RFC 2833	
102	Fax Bypass	20
103	Modem Bypass	20
104	RFC 2198 (Redundancy)	Same as channel's voice coder.
105	NSE Bypass	
114	MS RTA (WB)	20
115	MS RTA (NB)	20
76	SILK NB	20
77	SILK WB	20

8.3 Default RTP / RTCP / T.38 Port Allocation

The local default local User Datagram Protocol (UDP) ports for Audio and Fax media streams are calculated using the formula below:

```
BaseUDPPort + Channel ID * 10 + Port Offset
```

The BaseUDPPort is a configurable parameter, which by default is 6000. The port offsets are listed in the table below.

Table 8-3: Local UDP Port Offsets

Port Type	Port Offset
Audio RTP	0
Audio RTCP	1
Fax T.38	2

For example, the T.38 local UDP port for channel 30 is calculated as follows: 6000 + 30*10 + 2 = 6302.

Notes:

- For a description of the *ini* file parameter BaseUDPPort, refer to the device's *User's Manual*.
- To configure the device to use the same port for both RTP and T.38 packets, set the parameter T38UseRTPPort to 1.
- For the 2000 Series, the number of channels depends on the configuration (i.e., device with one or two TP-1610 / IPM-1610 blades.

Reader’s Notes

9 CAS Protocol Table

Note: This section is applicable only to Digital PSTN.

9.1 Constructing CAS Protocol Tables for CAS-Terminated Protocols

The protocol table file is a text file containing the protocol's state machine that defines the entire protocol process. It is constructed of States, predefined Actions/Events, and predefined functions. With this file, you have full control over CAS protocol and can define or modify any CAS protocol by writing the protocol state machine in a text file according to a few AudioCodes-defined rules.

➢ **To generate the protocol file:**

1. Learn the protocol text file rules from which the CAS state machine is built.
2. Refer to the supplied CAS files for an example.
3. Build the specific protocol/script text file (for example, xxx.txt) file and its related numerical value h file (for example, UserProt_defines_xxx.h). Note that the xxx.txt file must include the following 'C include' (for example, #include 'UserProt_defines_xxx.h').
4. Compile the xxx.txt with the 'TrunkPack Downloadable Conversion Utility' to produce the xxx.dat file. Note that the files xxx.txt, CASSetup.h, cpp.exe and UserProt_defines_xxx.h must be located in the same folder (you should choose Dynamic Format at the list).
5. Download the xxx.dat file to the board using the function acOpenBoard() in the initialization phase.

9.2 Protocol Table Elements

The *CASSetup.h* file includes all the predefined definitions necessary to build a new protocol text file or to modify an existing one.

The CAS protocol table file (xxx.txt) is composed of the following elements:

- INIT Variables
- Actions
- Functions
- States

9.2.1 INIT Variables

INIT variables are numeric values defined by users in UserProt_defines_xxx.h. These values can be used in the file xxx.txt.

For example, INIT_RC_IDLE_CAS defines the ABCD bits expected to be received in IDLE state. INIT_DTMF_DIAL defines the On-time and Off-time for the DTMF digits generated towards the PSTN. Refer to the detailed list in UserProt_defines_xxx.h and in the sample protocol text file (supplied CAS files). Refer to the following ST_INIT detailed explanation.

9.2.2 Actions

Actions (i.e., protocol table events) are protocol table events activated either by the DSP (e.g., EV_CAS_01) or by users (e.g., EV_PLACE_CALL, EV_TIMER_EXPIRED1). The full list of available predefined events is located in the file CASSetup.h.

9.2.3 Functions

Functions define a certain procedure that can be activated in any state or in the transition from one state to another. The available functions include, for example, SET_TIMER (timer number, timeout in milliseconds), SEND_CAS (AB value, CD value). A full list of the possible predefined functions can be found in the file CASSetup.h.

9.2.4 States

Each Protocol Table consists of several states that it switches between during the call setup and tear-down process. Every state definition begins with the prefix 'ST_' followed by the state name and colon. The body of the state is composed of up to four unconditional performed functions and a list of actions that may trigger this state.

Below shows an example taken from an E&M wink start table protocol file:

Table 9-1: ST_DIAL: Table Elements

Action	Function	Parameter		Next State
		#1	#2	
FUNCTION0	SET_TIMER	2	Extra Delay Before Dial	DO
EV_TIMER_EXPIRED2	SEND_DEST_NUM	ADDRESS	None	NO_STATE
EV_DIAL_ENDED	SET_TIMER	4	No Answer Time	ST_DIAL_ENDED

When the state machine reaches the dial state, it sets timer number 2 and then waits for one of two possible actions to be triggered: Either timer 2 expiration or end of dial event. When timer 2 expires, the protocol table executes function SEND_DEST_NUM and remains in the same state (NEXT_STATE=NO_STATE). When the dial event ends, the protocol table sets timer 4 and moves to ST_DIAL_ENDED written in the field NEXT_STATE.

Although you can define your own states, there are two states defined in the file *CASSetup.h* that must appear in every protocol table created:

- **ST_INIT:** When channels initialization is selected, the table goes into ‘Init’ state. This state contains functions that initialize the following global parameters:
  - **INIT_RC_IDLE_CAS:** Defines the ABCD bits expected to be received in the IDLE state in the specific protocol. The third parameter used to enable detection of 4 bits` CAS value (see below).

    - **INIT_TX_IDLE_CAS:** Defines the ABCD bits transmitted in IDLE state in the specific protocol.
    - **INIT_DIAL_PLAN:** A change regarding the issue of an incoming call dialed number. In version 4.2 and earlier, users were required to predefine the expected number of digits to receive an incoming call. If a lower number of digits than expected was received, the call setup would have failed.
- **ST_IDLE:** When no active call is established or is in the process of being established, the table resides in Idle state, allowing it to start the process of incoming or outgoing calls. When the call is cleared, the state machine table returns to its Idle state.

In Versions 4.2 and later, process the incoming call detection event by declaring end of digit reception in the following ways (both for ADDRESS/destination number and ANI/source number):

- Receiving '#' digit (in MF or DTMF).
- The number of digits collected reaches its maximum value as defined in DIAL_PLAN parameter #1 and #2 for destination and ANI numbers respectively.
- A predefined time-out value defined in DIAL_PLAN parameter #3 elapses.
- In MFC-R2 reception of signal I-15 (depending on the variant).

Parameter	Description
INIT_DTMF_DIAL	Defines the On-time and Off-time for the DTMF digits generated towards the PSTN.
INIT_COMMA_PAUSE_TIME	Defines the delay between each digit when a comma is used as part of the dialed number string (see acPSTNPlaceCall for details).
INIT_DTMF_DETECTION	Defines the minimum/maximum On-time for DTMF digit dialing detection.
INIT_PULSE_DIAL_TIME	Not supported by the current stack version. Defines the Break and Make time for pulse dialing.
INIT_PULSE_DIAL	Not supported by the current stack version. Defines the Break and Make ABCD bits for pulse dialing.
INIT_DEBOUNCE	Defines the interval time of CAS to be considered (a stable one).
INIT_COLLECT_ANI	Enables or Disables reception of ANI in a specific protocol.
INIT_DIGIT_TYPE	The #1 parameter defines the dialing method used (DTMF, MF). With MFC-R2 protocols, this parameter is not applicable (digits are assumed to be R2 digits). The #2 parameter enabled to usage of SS5 tones (not used). The #3 parameter used to enable digits detection at the OutGoing side of the call (which needed at some protocols.
INIT_NUM_OF_EVENT_IN_STATE	Inserted for detection on TOTAL_NUMBER_OF_EVENTS_IN_STATE (CASSetup.h).

Parameter	Description
INIT_INIT_GLOBAL_TIMERS	Initiates specific timers; it is used with Parameter#1 for metering pulse timer duration.
INIT_PULSE_DIAL_ADDITIONAL_PARAMS	Not used.
INIT_RINGING_TO_ANALOGUE	When using analogue gateway option, it defines the CAS value of ringing (#1) CAS value of silence (#2) and CAS value of polarity relevsal(#3).
INIT_DIGIT_TYPE_1	Defines the signaling system used to send operator service.
INIT_REJECT_COLLECT	Defines the method for reject collect calls: *disabled*, *using Line signaling,* or *using register signaling*.
INIT_VERSION	Defines the version number. The version number is relevant to the release version number and is a text information string (not related to the utility compilation version number).
INIT_SIZE_OF_TABLE_PARAM	Users must insert the definition of TOTAL_NUMBER_OF_EVENTS_IN_STATE from CASSetup.h.

9.3 Reserved Words

For reserved words such as DO, NO_STATE, etc., see the detailed list in CASSetup.h.

9.4 State Line Structure

Each text line in the body of each state comprises 6 columns:

1. Action/event
2. Function
3. Parameter #1
4. Parameter #2
5. Additional parameters
6. Next state

9.5 Action / Event

Action / event is the name of the table's events that are the possible triggers for the entire protocol state machine. These can be selected from the list of events in file CASSetup.h (e.g., EV_DISCONNECT_INCOMING).

At the beginning of the state, there can be up to four unconditional actions / events called FUNCTION. These events are functions that are unconditionally performed when the table reaches the state. These actions are labeled FUNCTION0 to FUNCTION3.

The following subsections provide a list of available protocols table actions (events to the state machine).

9.5.1 User Command Oriented Action / Event

Table 9-2: User Command Orientated Action / Event

User Command Oriented Action/Event	Description
EV_PLACE_CALL	When acpstnplacecall() is used.
EV_SEIZE_LINE	Used by Megaco control protocol.
EV_SEND_SEIZE_ACK	Used by Megaco control protocol.
EV_ANSWER	When acpstnanswercall() is used.
EV_MAKE_DOUBLE_ANSWER_CAS	When the function acpstnanswercall is used and the INIT_REJECT_COLLECT parameter is set to Line Signaling.
EV_MAKE_DOUBLE_ANSWER_MF	When the function acpstnanswercall is used and the INIT_REJECT_COLLECT parameter is set to Register Signaling.
EV_DISCONNECT	When function acpstndisconnectcall() is used and the call is outgoing.
EV_DISCONNECT_INCOMING	When function acpstndisconnectcall() is used and the call is incoming.
EV_RELEASE_CALL	When acpstnreleasecall() is used.
EV_FORCED_RELEASE	When accasforcedrelease () is used.
EV_USER_BLOCK_COMND	When accasblockchannel() is used. This event is used to block or unblock the channel.
EV_MAKE_METERING_PULSE	When the function accasmeteringpulse is used, it triggers the start of the metering pulse while using function set_pulse_timer to start the timer to get the off event (see event ev_metering_timer_pulse_off).
EV_METERING_TIMER_PULSE_OFF	An event sent after the timer (invoked by function set_pulse_timer) expires. Refer to ev_make_metering_pulse.
EV_MAKE_FLASH_HOOK	When accasflashhook is used, a flash hook is triggered.

9.5.2 CAS Change Oriented Events

Table 9-3: CAS Change Orientated Events

Event	Description
EV_CAS_1_1	A new cas a, b bits received (a=1, b=1, was stable for the bouncing period).
EV_CAS_1_0	A new cas a, b bits received (a=1, b=0, was stable for the bouncing period).
EV_CAS_0_1	A new cas a, b bits received (a=0, b=1, was stable for the bouncing period).

Event	Description
EV_CAS_0_0	A new cas a, b bits received (a=0, b=0, was stable for the bouncing period).
EV_CAS_1_1_1_1	A new cas a, b bits received (a=1, b=1, c=1, d=1 was stable for the bouncing period). To receive such detection (that is different from EV_CAS_1_1) you must set YES at the #3 parameter of INIT_RC_IDLE_CAS.

9.5.3 Timer Oriented Events

Table 9-4: Time-Orientated Events

Event	Description
EV_TIMER_EXPIRED1	Timer 1 that was previously set by the table has expired.
EV_TIMER_EXPIRED2	Timer 2 that was previously set by the table has expired.
EV_TIMER_EXPIRED3	Timer 3 that was previously set by the table has expired.
EV_TIMER_EXPIRED4	Timer 4 that was previously set by the table has expired.
EV_TIMER_EXPIRED5	Timer 5 that was previously set by the table has expired.
EV_TIMER_EXPIRED6	Timer 6 that was previously set by the table has expired.
EV_TIMER_EXPIRED7	Timer 7 that was previously set by the table has expired.
EV_TIMER_EXPIRED8	Timer 8 that was previously set by the table has expired.

9.5.4 Counter Oriented Events

Table 9-5: Counter Orientated Events

Event	Description
EV_COUNTER1_EXPIRED	The value of counter 1 reached 0.
EV_COUNTER2_EXPIRED	The value of counter 2 reached 0.

9.5.5 IBS Oriented Events

Table 9-6: IBS Orientated Events

Event	Explanation
EV_RB_TONE_STARTED	Ringback tone as defined in the Call Progress Tone *ini* file (type and index) is detected.
EV_RB_TONE_STOPPED	Ringback tone as defined in the Call Progress Tone *ini* file (type and index) is stopped after it was previously detected.
EV_BUSY_TONE	Not used.

Event	Explanation
EV_BUSY_TONE_STOPPED	Not used.
EV_FAST_BUSY_TONE	Not used.
EV_FAST_BUSY_TONE_STOPPED	Not used.
EV_ANI_REQ_TONE_DETECTED	R1.5 ANI-requset tone as defined in the Call Progress Tone *ini* file (type and index) is detected.
EV_R15_ANI_DETECTED	R1.5 ANI digit-string was detected.
EV_DIAL_TONE_DETECTED	Dial tone as defined in the Call Progress Tone *ini* file (type and index) is detected.
EV_DIAL_TONE_STOPPED	Dial tone as defined in the Call Progress Tone *ini* file (type and index) is stopped after it was previously detected.

9.5.6 DTMF/MF Oriented Events

Table 9-7: DTMF / MF Orientated Events

Event	Explanation
EV_MFRn_0	MF digit 0 is detected (only DTMF & MFr1).
EV_MFRn_1	MF digit 1 is detected.
EV_MFRn_2	MF digit 2 is detected.
EV_MFRn_3	MF digit 3 is detected.
EV_MFRn_4	MF digit 4 is detected.
EV_MFRn_5	MF digit 5 is detected.
EV_MFRn_6	MF digit 6 is detected.
EV_MFRn_7	MF digit 7 is detected.
EV_MFRn_8	MF digit 8 is detected.
EV_MFRn_9	MF digit 9 is detected.
EV_MFRn_10	MF digit 10 is detected.
EV_MFRn_11	MF digit 11 is detected.
EV_MFRn_12	MF digit 12 is detected.
EV_MFRn_13	MF digit 13 is detected.
EV_MFRn_14	MF digit 14 is detected.
EV_MFRn_15	MF digit 15 is detected.
EV_MFRn_1_STOPPED	MF digit 1 previously detected is now stopped.
EV_MFRn_2_ STOPPED	MF digit 2 previously detected is now stopped.
EV_MFRn_3_ STOPPED	MF digit 3 previously detected is now stopped.
EV_MFRn_4_ STOPPED	MF digit 4 previously detected is now stopped.
EV_MFRn_5_ STOPPED	MF digit 5 previously detected is now stopped.

Event	Explanation
EV_MFRn_6_ STOPPED	MF digit 6 previously detected is now stopped.
EV_MFRn_7_ STOPPED	MF digit 7 previously detected is now stopped.
EV_MFRn_8_ STOPPED	MF digit 8 previously detected is now stopped.
EV_MFRn_9_ STOPPED	MF digit 9 previously detected is now stopped.
EV_MFRn_10_ STOPPED	MF digit 10 previously detected is now stopped.
EV_MFRn_11_ STOPPED	MF digit 11 previously detected is now stopped.
EV_MFRn_12_ STOPPED	MF digit 12 previously detected is now stopped.
EV_MFRn_13_ STOPPED	MF digit 13 previously detected is now stopped.
EV_MFRn_14_ STOPPED	MF digit 14 previously detected is now stopped.
EV_MFRn_15_ STOPPED	MF digit 15, previously detected is now stopped.
EV_END_OF_MF_DIGIT	When DialMF() is used and no more dialed number digits are available (they already were sent). For example, the far side requests the next ANI digit but all digits already have been sent. This event usually appears in MFC-R2 tables.
EV_FIRST_DIGIT	The first digit of the DNI / ANI number is detected.
EV_DIGIT_IN	An incoming digit (MFR1 or DTMF) is detected.
EV_WRONG_MF_LENGTH	An incoming digit was detected, but its duration (ON-TIME) is too long or too short.
EV_DIALED_NUM_DETECTED	The whole destination number is detected.
EV_ANI_NUM_DETECTED	The whole source number is detected.
EV_DIAL_ENDED	The dialing process finished and all digits dialed.
EV_NO_ANI	When DialMF() is used and no ANI is specified by the outgoing user in function acPSTNPlaceCall(). MFC

Note: MF digit includes MF R1, R2-FWD, or R2-BWD, according to the context, protocol type, and call direction.

The following actions / events cause the MFC-R2 table to send the correct MF tone to the backward direction:

Table 9-8: Actions / Events Causing MFC-R2 Table to Send Correct MF Tone to Backward Direction

Actions/Events	Explanation
EV_ACCEPT	When acCASAcceptCall is used (only in MFC-R2) with CALLED_IDLE as its reason parameter (for example, this sends MF backward B-6).
EV_ACCEPT_SPARE_MF1	When acCASAcceptCall is used with SPARE_MF1 as its reason parameter.
EV_ACCEPT_SPARE_MF9	When acCASAcceptCall is used with SPARE_MF9 as its reason parameter.

Actions/Events	Explanation
EV_ACCEPT_SPARE_MF10	When acCASAcceptCall is used with SPARE_MF10 as its reason parameter.
EV_ACCEPT_SPARE_MF11	When acCASAcceptCall is used with SPARE_MF11 as its reason parameter.
EV_ACCEPT_SPARE_MF12	When acCASAcceptCall is used with SPARE_MF12 as its reason parameter.
EV_ACCEPT_SPARE_MF13	When acCASAcceptCall is used with SPARE_MF13 as its reason parameter.
EV_ACCEPT_SPARE_MF14	When acCASAcceptCall is used with SPARE_MF14 as its reason parameter.
EV_ACCEPT_SPARE_MF15	When acCASAcceptCall is used with SPARE_MF 15 as its reason parameter.
EV_REJECT_BUSY	When acCASAcceptCall is used with CALLED_BUSY as its reason parameter.
EV_REJECT_CONGESTION	When acCASAcceptCall is used with CALLED_CONGESTION as its reason parameter.
EV_REJECT_UNALLOCATED	When acCASAcceptCall is used with CALLED_UNALLOCATED as its reason parameter.
EV_REJECT_SIT	When acCASAcceptCall is used with SIT as its reason parameter.
EV_REJECT_RESERVE1	When acCASAcceptCall is used with CALLED_RESERVE1 as its reason parameter.
EV_REJECT_RESERVE2	When acCASAcceptCall is used with CALLED_RESERVE2 as its reason parameter.

9.5.7 Operator Service Events (up to GR-506)

Table 9-9: Operator Service Events (Up to GR-506)

Event	Explanation
EV_SEND_LINE_OPERATOR_SERVICE1	Send operator service 1 (=Operator Released) using line signaling.
EV_SEND_LINE_OPERATOR_SERVICE2	Send operator service 2 (=Operator Attached) using line signaling.
EV_SEND_LINE_OPERATOR_SERVICE3	Send operator service 3 (=Coin Collect) using line signaling.
EV_SEND_LINE_OPERATOR_SERVICE4	Send operator service 4 (=Coin Return) using line signaling.
EV_SEND_LINE_OPERATOR_SERVICE5	Send operator service 5 (=Ring-back) using line signaling.
EV_SEND_REGISTER_OPERATOR_SERVICE1	Send operator service 1 (=Operator Released) using register signaling.

Event	Explanation
EV_SEND_REGISTER_OPERATOR_SERVICE2	Send operator service 2 (=Operator Attached) using register signaling.
EV_SEND_REGISTER_OPERATOR_SERVICE3	Send operator service 3 (=Coin Collect) using register signaling.
EV_SEND_REGISTER_OPERATOR_SERVICE4	Send operator service 4 (=Coin Return) using register signaling.
EV_SEND_REGISTER_OPERATOR_SERVICE5	Send operator service 5 (=Ring-back) using register signaling.
EV_SEND_REGISTER_OPERATOR_SERVICE6	Send operator service 6 (=Coin Collect/Operator Released) using register signaling.

Note: The following actions/events are for internal use only:

- EV_INIT_CHANNEL
- EV_TO_USER
- EV_CLOSE_CHANNEL
- EV_OPEN_CHANNEL
- EV_FAIL_DIAL
- EV_FAIL_SEND_CAS
- EV_ALARM

9.6 Function

The function's column holds the name of the function to be activated when the action specified in the action / events field occurs. Select the functions from the list of eight functions defined in CasSetup.h (e.g., START_COLLECT). When NONE is specified in this column, no function is executed.

Note: Do not define the same timer number (by SET_TIMER) twice before the first one expires or is deleted.

9.7 Parameters

The following columns are used as the function's parameters:

- **Parameter #1**
- **Parameter #2**

The list of global parameters can be found in CasSetup.h. If a parameter is not essential, it can also be written as NONE.

Note: In previous versions, you must include three parameters per function. From Release 5.2 and on, to enable the dynamic format of the CAS file and reduce memory usage, you can only include the used parameters.

Table 9-10: Available User Functions and Corresponding Parameters

User Function	User Function Parameters and Descriptions
SET_TIMER	(Timer number, timeout). Sets the timers managed per B-channel. Their expiration triggers the state machine table. Each protocol table/state machine can use up to 8 timers per B-channel/call (timeout in msec) when the timers have 25 msec resolution.
SEND_CAS	(AB value, CD value). ABCD bits are sent as line signaling for the specific channel when the call is setup.
GENERATE_CAS_EV	Check the ABCD bits value, and send a proper event to the state machine.
SEND_EVENT	(Event type, cause). The specific event type is sent to the host/user and retrieved by applying the function acGetEvent().
SEND_DEST_NUM	En-bloc dialing: refers to the digits string located in function acPSTNPlaceCall. Three types are available: (1) DestPhoneNum (2) InterExchangePrefixNum (3) SourcePhoneNum.
DEL_TIMER	(Timer number). Deletes a specific timer or all the timers (0 represents all the timers) for the B-channel.
START_COLLECT	Initiates the collection of address information, i.e., the dialed (destination) number for incoming calls where appropriate, according to the protocol. In the time between START_COLLECT and STOP_COLLECT, no digit is reported to users (EV_DIGIT is blocked) and the destination number is reported in event EV_INCOMING_CALL_DETECTED.
STOP_COLLECT	Refer to START_COLLECT.
SET_COUNTER	(Counter number, counter value or NONE). Sets counters managed per B-channel. Their expiration triggers the state machine. The counter initialization value should be a non-negative number. To delete all timers, invoke this function with 0 in the counter number field.
DEC_COUNTER	(Counter number). Decreases the counter value by 1. When the counter value reaches 0, EV_COUNTERx_EXPIRES is sent to the table (where x represents the counter number).
RESTRICT_ANI	Indicate the incoming side to hide the ANI from the Far-end user.
SEND_MF	(MF type, MF digit or index or NONE, MF sending time). This function is used only with MFC-R2 protocols.

The Channel Parameter structure contains three parameters associated with sending digits:

Table 9-11: Parameters Associated with Sending Digits

Parameter	Description
AddressVector and ANIDigitVector	These parameters are initialized when function PlaceCall is used. When the code reaches the dialing section, it sends the MF digit according to the MF type specified in the MF type cell (the types are defined in file CASSetup.h):

Parameter	Description
	▪ **ADDRESS:** Sends the digit from the address vector (destination number) according to the index requested. Refer to the Index definition. ▪ **ANI:** Sends the digit from the ANI vector (source number) according to the requested index. ▪ **SPECIFIC:** Sends the MF digit specified in the cell Parameter #2. ▪ **SOURCE_CATEGORY:** Sends the predefined source category MF digit. The source category digit is set as the parameter SourceNumberingType when function PlaceCall is used. The second and third parameters are ignored when this type is used. ▪ **TRANSFER_CAPABILITY:** Sends the predefined line category MF digit. The line category digit is set as the parameter TransferCapability when function PlaceCall is used. The second and third parameters are ignored when this type is used.
Index	Specifies the Offset of the next digit to be sent from the vector (ADDRESS or ANI types, described above): ▪ **Index 1:** Sends the next digit in the vector. ▪ **Index –n: S**ends the last n digit. Underflow can occur if n is greater than the number of digits sent so far. ▪ **Index 0:** Sends the last sent digit. ▪ **Index SEND_FIRST_DIGIT:** Starts sending the digits vector from the beginning (see CASSetup.h).
MF Send Time	This send time parameter specifies the maximum transmission time of the MF. ▪ **STOP_SEND_MF:** Stops sending the current MF. ▪ **SEND_PROG_TON:** Operation, Tone or NONE.

Two operations are available:

- Sends the Call Progress Tone specified in the cell Parameter #2 (The second parameter can be taken from CASsetup.h)
- Stops sending the last parameter

Parameter	Description
CHANGE_COLLECT_TYPE	(Collect Type). Used by the incoming user to indicate that waiting for receipt of the digit of the requested type. The type can be one of the following: ▪ **ADDRESS:** The user waits for receipt of address digits. ▪ **ANI:** The user waits for receipt of ANI digits. ▪ **SOURCE_CATEGORY:** The user waits for receipt of the source category. ▪ **TRANSFER_CAPABILITY:** The user waits for receipt of the source transfer capability (line category).

9.8 Next State

The Next State column contains the next state the table moves to after executing the function for that action/event line. When you select to stay in the same state, insert NO_STATE or use the current state.

Note the difference between NO_STATE and the current state name in this field. If you select to stay in the same current state, the unconditional actions (FUNCTION0) at the beginning of the state are performed. In contrast, NO_STATE skips these functions and waits for another action to arrive.

Reserved word 'DO' must be written in the next state field if the unconditional actions (FUNCTION0) at the beginning of the state are used.

9.9 Changing the Script File

- CAS bouncing is filtered globally for each received CAS for each channel. Define the time for the filtering criteria in the protocol table file (see INIT_DEBOUNCE) and this exceeds the bouncing in the DSP detection of 30 msec.
- ANI/CLI is enabled using parameter ST_INIT ANI with 'YES'. ANI/CLI is supported using EV_ANI_NUM_DETECTED as the table action for collecting the ANI number in an incoming call. For outgoing calls, the table's function SEND_DEST_NUM with ANI parameter I initiates ANI dialing. The ANI number is provided by you in the Source phone number parameter of acPSTNPlaceCall().
- You can use ANSI C pre-compile flags such as #ifdef, #ifndef, #else and #endif in the CAS script file. For example, you can decide whether or not to play dial tone according to fulfillment of #ifdef statement. The definition itself must be in CASSetup.h.

9.9.1 MFC-R2 Protocol

- Use the SEND_MF script function to generate the outgoing call destination number. In this case, the first parameter should be ADDRESS (or ANI for source phone number) and the second parameter –3 to 1 (+1), indicating which digit is sent out of the number that the string conveyed by you in acPSTNPlaceCall().
    - 1 (+1) implies sending of the next digit
    - 0 implies a repeat of the last digit
    - -1 implies the penultimate digit
      This parameter actually changes the pointer to the phone number string of digits. Thus, a one-to-one mapping with the MF backward signals of the R2 protocol exists.
- Using parameter SEND_FIRST_DIGIT initiates resending the string from the beginning, (change the pointer back to first digit and then proceed as above). This parameter is defined in CASSetup.h.
- When MFC-R2 protocol is used, the two detectors (opened by default) are the Call Progress Tones and MFC-R2 Forward MF. When you invoke an outgoing call via acPSTNPlaceCall(), MFC-R2 Forward MF detector is replaced with MFC-R2 Backward MF detector, since only two detectors per DSP channel are permitted to operate simultaneously.
- The correct MF is automatically generated according to the call direction: Forward for outgoing calls and Backward for incoming calls.

- MFC-R2 protocol fault can cause a channel block. In this case, the script file (supplied) releases the call to enable the user to free the call resources and be notified as to being in blocking state.
- START_COLLECT and STOP_COLLECT must be used in the script file for MF collecting both in outgoing and incoming calls.

> **Warning:** If this script function isn't used, the script gets stuck and forward\backward MF are not detected.

- The Ringback Call Progress Tone is translated to a unique event acEV_PSTN_ALERTING, since the Ringback tone is actually used in all AudioCodes protocols' state machines. All other Call Progress Tones are conveyed via acEV_TONE_DETECTED and retrieved by the user according to their type and index (note that the Ringback tone should be defined in the Call Progress Tones table with the relevant type in order to get this event).
- When the tone detection event is received, users can perform any action. For example, if the event is received with BUSY tone indication, users can invoke acPSTNDisconnectCall() to end the call.
- The MFC-R2 destination number is collected using parameter EXPECTED_NUM_OF_DIGITS_MINUS_1 for SET_COUNTER that the user defines with UserProt_defines_R2_MF.h. The counter function is used to trigger the script file for the penultimate received. After receiving the last digit, the script file (acting as the outgoing register) initiates the A6/A3 FWD MF. Normally, variant supports end of digit information (MF15 or MF12) or silence at the end of the dialing (when MF15 is not used). A short pulse of MF3 (A3) is sent to indicate that the entire string of digits (according to Q442, 476) is received.
- Sending Group B digit by an incoming register requires invoking acCASAcceptCall() with a certain reason parameter. Six reason parameters are available:

Reason Parameter	Description
CALLED_IDLE	Subscriber's line is free. Continue the call sequence. Should usually be followed by accept or reject.
CALLED_BUSY	Subscriber line is busy. Perform disconnect procedures.
CALLED_CONGESTION	Congestion encountered. Perform disconnect procedures.
CALLED_UNALLOCATED	Dial number was not allocated. Perform disconnect procedures.
CALLED_RESERVE1	Reserved for additional group B (user additional requirements).
CALLED_RESERVE2	Reserved for additional group B (user additional requirements).

Each reason generates a specific action, defined by the user, who modifies the script file. The action is then used to generate/respond with a group B MF (free, busy, etc.).

- **Transfer Capability:** This parameter under function acPSTNPlaceCAll() is used by the outgoing register to generate the service nature of the originating equipment. In most variants (countries), this is the same as the Calling Subscriber Categories, but in some countries it is different, such as in R2 China protocol where it is referred to as the KD (Group II) digit.

Note: This parameter only receives MF values from the enumerator acTISDNTransferCapability. Choose the MF digit according to the service type that should be sent.

- **Source Category:** This parameter under function acPSTNPlaceCall() determines the calling subscriber category. For example, a subscriber with priority, a subscriber without priority, etc. The parameter is usually sent as part of the Group II forward digits (except for R2 China where it is sent as the KA digit using Group I forward digits).

Note: This parameter is applicable only only to MFC-R2 protocol type.

Reader’s Notes

10 Accessory Utilities

AudioCodes provides various utilities that provide you with user-friendly interfaces that enhance device usability and facilitate your transition to the new VoIP infrastructure. The following proprietary applications are available:

- **BootP / TFTP Server** configuration utility - see "BootP/TFTP Configuration Utility" on page 183
- **TrunkPack Downloadable Conversion Utility** (DConvert) - see "TrunkPack Downloadable Conversion Utility" on page 195
- **Call Progress Tones Wizard** - see "Call Progress Tones Wizard" on page 209

10.1 BootP/TFTP Server Configuration Utility

Notes:

- The utility supports up to 200 clients (MAC entries).
- BootP is applicable to Mediant 800 MSBG and Mediant 1000 MSBG only for recovering these devices from "rescue" mode.

The BootP/TFTP Server utility is AudioCodes proprietary BootP / TFTP server (hereafter referred to as *utility*). The utility comprises two functionalities:

- **Bootstrap Protocol (BootP):**
  - Assigns the device networking parameters (such as IP address, subnet mask, and default gateway).
  - Provides the device with the TFTP server's IP address (and the *ini* and cmp file names) from where the device can load these files via TFTP.

  Complies to BootP standards RFC 951 and RFC 1542

- **Trivial File Transfer Protocol (TFTP):** The TFTP server functionality allows you to load the device with the following files from a TFTP server:
  - Firmware file (*. *cmp*) for upgrading the device
  - Configuration file (*.*ini* file) for modifying the device's configuration settings
  - Auxiliary files (such as call progress tones / CPT)

  Complies to TFTP standards RFC 1350 and RFC 906

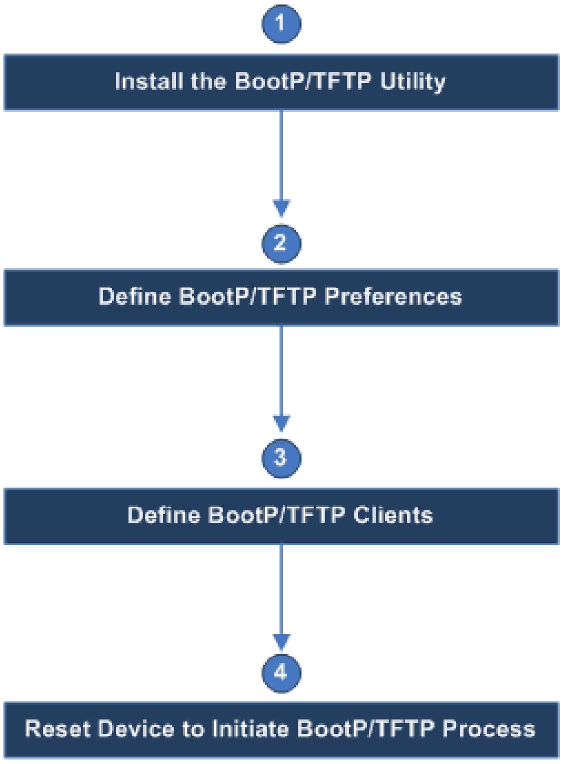
When the device powers up (or is physically reset), it broadcasts a BootRequest message on the network (no BootP message is sent when the device is reset from the Web interface). A BootP server on the network receives this message and generates a BootReply if the device is successfully identified (according to its MAC address). The BootReply indicates the networking parameters that must be used by the device and optionally, specifies the *ini* and cmp file names and IP address of the TFTP server (from where these files must be loaded).

Therefore, the utility can be used for initializing the device, by providing it initial networking parameters. However, the utility is also useful for restoring connectivity to the device if lost. This loss of connectivity can be due to, for example, any of the following reasons:

- Software upgrade failure (when done through the Web interface)
- IP address no longer known (due to whatever reason) – device obtains new IP address from BootP
- Web interface has been inadvertently disabled
- Web interface's password is no longer known (due to whatever reason)
- Device has encountered a fault that cannot be recovered using the Web interface

The steps for configuring and using the utility are summarized in the flowchart below:

Figure 10-1: Flowchart for using the BootP / TFTP Server Utility


1
Install the BootP/TFTP Utility
2
Define BootP/TFTP Preferences
3
Define BootP/TFTP Clients
4
Reset Device to Initiate BootP/TFTP Process

10.1.1 Installing the BootP / TFTP Server Utility

The utility can be installed on the following Windows™ operating systems (OS):

- Windows NT
- Windows 2000
- Windows XP
- Windows Vista and Windows 7

Notes:

- The BootP / TFTP utility cannot be installed on other OS, such as Linux.
- The BootP / TFTP utility cannot be installed on 64-bit Windows OS.

➢ To install the utility on a computer:

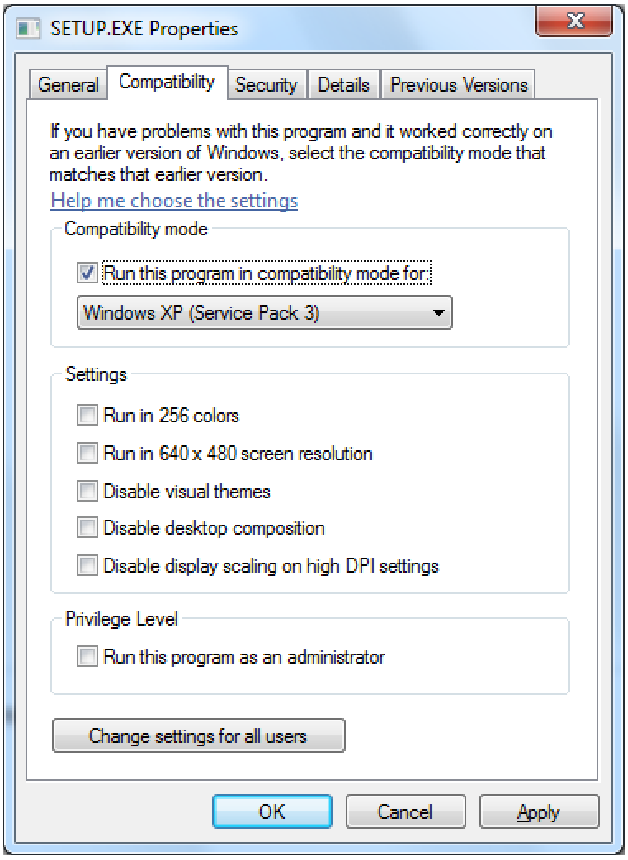
1. Download the utility's installation file from AudioCodes Web site:
   a. Browse to http://www.audiocodes.com/downloads and then login as a registered customer.
   b. Use the available drop-down boxes to search for the required software files.
   c. Click the **BootP & TFTP Configuration utility.zip** link, select the 'I accept' check box, click **DOWNLOAD**, and then save the file to a folder on your PC.

Note: You must be a registered customer with login credentials to download this file from the Web site.

2. Unzip the downloaded file.
3. If you are installing on a Windows Vista or Windows 7 machine, do the following (otherwise, skip to Step 4):
   a. Right-click the *SETUP.exe* file, and then from the shortcut menu, choose **Properties**.
   b. Select the **Compatibility** tab.

c. Select the 'Run this program in compatibility mode for' option, and then from the drop-down list, select **Windows XP (Service Pack 3)**, as shown below.

Figure 10-2: Compatibility Tab (Windows Vista or Windows 7)



SETUP.EXE Properties

If you have problems with this program and it worked correctly on an earlier version of Windows, select the compatibility mode that matches that earlier version.

Help me choose the settings

Compatibility mode

☑ Run this program in compatibility mode for:

Windows XP (Service Pack 3)

Settings

☐ Run in 256 colors

☐ Run in 640 x 480 screen resolution

☐ Disable visual themes

☐ Disable desktop composition

☐ Disable display scaling on high DPI settings

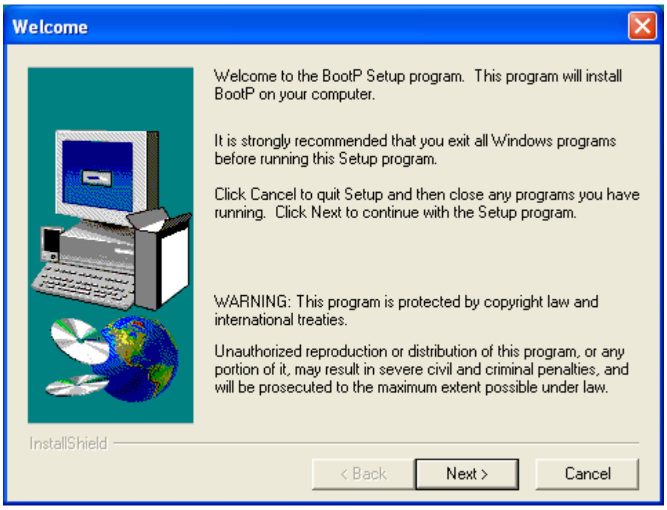
Privilege Level

☐ Run this program as an administrator

d. Click **OK**.

4. Click the *SETUP.exe* file to start the installation wizard (shown below).

Figure 10-3: BootP/TFTP Utility Installation Wizard



Welcome

Welcome to the BootP Setup program. This program will install BootP on your computer.

It is strongly recommended that you exit all Windows programs before running this Setup program.

Click Cancel to quit Setup and then close any programs you have running. Click Next to continue with the Setup program.

WARNING: This program is protected by copyright law and international treaties.

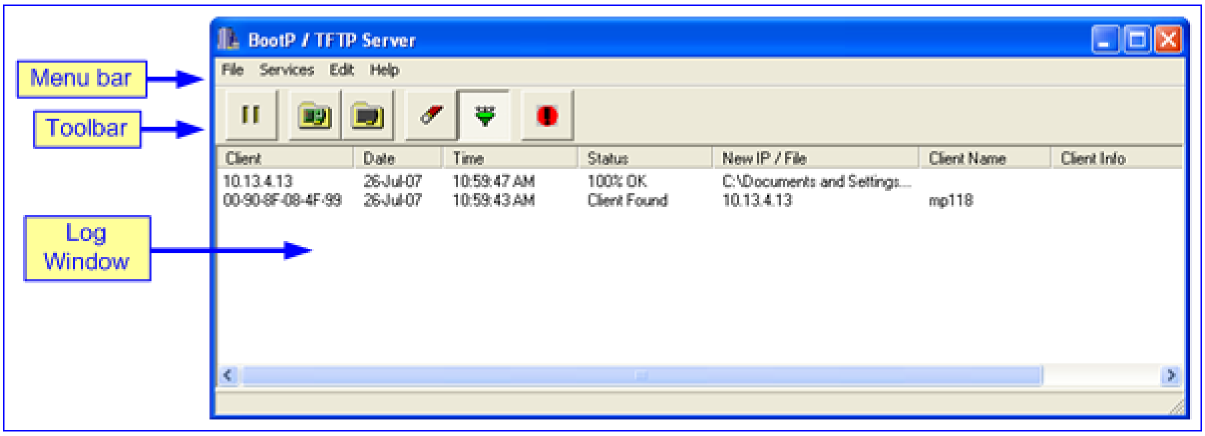
Unauthorized reproduction or distribution of this program, or any portion of it, may result in severe civil and criminal penalties, and will be prosecuted to the maximum extent possible under law.

5. Follow the wizard's instructions to install the utility.

10.1.2 BootP/TFTP Utility Main Window Description

The figure below shows the main window of the BootP/TFTP utility.

Figure 10-4: Main Screen



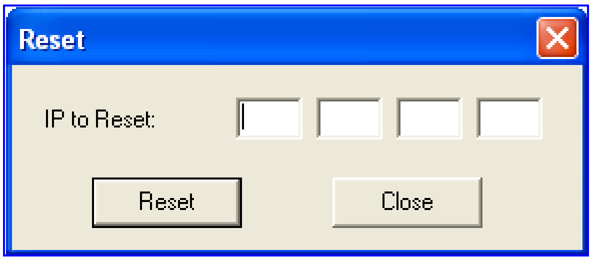
Client	Date	Time	Status	New IP / File	Client Name	Client Info
10.13.4.13	26-Jul-07	10:59:47 AM	100% OK	C:\Documents and Settings...		
00-90-8F-08-4F-99	26-Jul-07	10:59:43 AM	Client Found	10.13.4.13	mp118	

10.1.2.1 Toolbar Buttons in the Main Screen

The buttons on the toolbar are described in the table below:

Button	Name	Description
⏸	**Pause**	Pauses the BootP / TFTP utility so that no replies are sent to BootP requests. Click the button again to restart the BootP utility so that it responds to all BootP requests. The **Pause** button provides a depressed graphic when the feature is active.
📁	**Edit Clients**	Opens the Client Configuration window that enables you to enter configuration information for each supported device. Details on the Client Configuration window are provided in "Configuring the BootP Clients" on page 192.
📁	**Edit Templates**	Opens the Templates window that enables you to create or edit standard templates. These templates can be used when configuring new clients that share most of the settings. Details on the Templates window are provided in "Managing Client Templates" on page 194.
✎	**Clear Log**	Clears all entries from the Log window portion of the main window. Details on the Log window are provided in "Log Window" on page 188.
▼	**Filter Unknown Clients**	Prevents the BootP / TFTP utility from logging BootP requests received from disabled clients or from clients which do not have entries in the Clients table.
●	**Reset**	Opens the Reset window where you enter an IP address requests for a device that you want to reset. Refer to the figure below.

Figure 10-5: Reset Screen



When a device resets, it first sends a BootRequest. Therefore, the Reset button can be used to force a BootP session with a device without needing to power cycle the device. As with any BootP session, the computer running the BootP tool must be located in the same subnet as the controlled device.

10.1.2.2 Log Window

The Log window (see "BootP/TFTP Application User Interface" on page 187) records all BootP request and BootP reply transactions, as well as TFTP transactions. For each transaction, the Log window displays the following information:

- **Client:** address of the device, which is the MAC address of the client for BootP transactions or the IP address of the client for TFTP transactions.
- **Date:** date of the transaction, based on the internal calendar of the computer.
- **Time:** time of day of the transaction, based on the internal clock of the computer.
- **Status:** status of the transaction:
  - *Client Not Found:* A BootRequest was received but there is no matching client entry in the BootP / TFTP utility.
  - *Client Found:* A BootRequest was received and there is a matching client entry in the BootP / TFTP utility. A BootReply is sent.
  - *Client's MAC Changed:* There is a client entered for this IP address but with a different MAC address.
  - *Client Disabled:* A BootRequest was received and there is a matching client entry in the BootP / TFTP utility, but this entry is disabled.
  - *Listed At:* Another BootP utility is listed as supporting a particular client when the **Test Selected Client** button is clicked (for details on Testing a client, see "Testing the Client" on page 194).
  - *Download Status:* Progress of a TFTP load to a client, shown in %.
- **New IP / File:** IP address applied to the client as a result of the BootP transaction as well as the file name and path of a file transfer for a TFTP transaction.
- **Client Name:** client name as configured for that client in the Client Configuration window.

Right-click a row in the Log window to open a pop-up window with the following options:

- **Reset:** Selecting this option results in a reset command being sent to the client device. The program searches its database for the MAC address indicated in the line. If the client is found in that database, the program adds the client MAC address to the Address Resolution Protocol (ARP) table of the computer. The program then sends a reset command to the client. This enables a reset to be sent without knowing the current IP address of the client as long as the computer sending the reset is on the

same subnet.
Note: To use reset, you must have administrator privileges on the computer. Attempting to perform this type of reset without administrator privileges on the computer results in an error message. **ARP Manipulation Enable** must also be turned on in the Preferences window.

- **View Client:** Selecting this option, or double clicking on the line in the log window, opens the Client Configuration window. If the MAC address indicated on the line exists in the client database, it is highlighted. If the address is not in the client database, a new client is added with the MAC address filled out. You can enter data in the remaining fields to create a new client entry for that client.

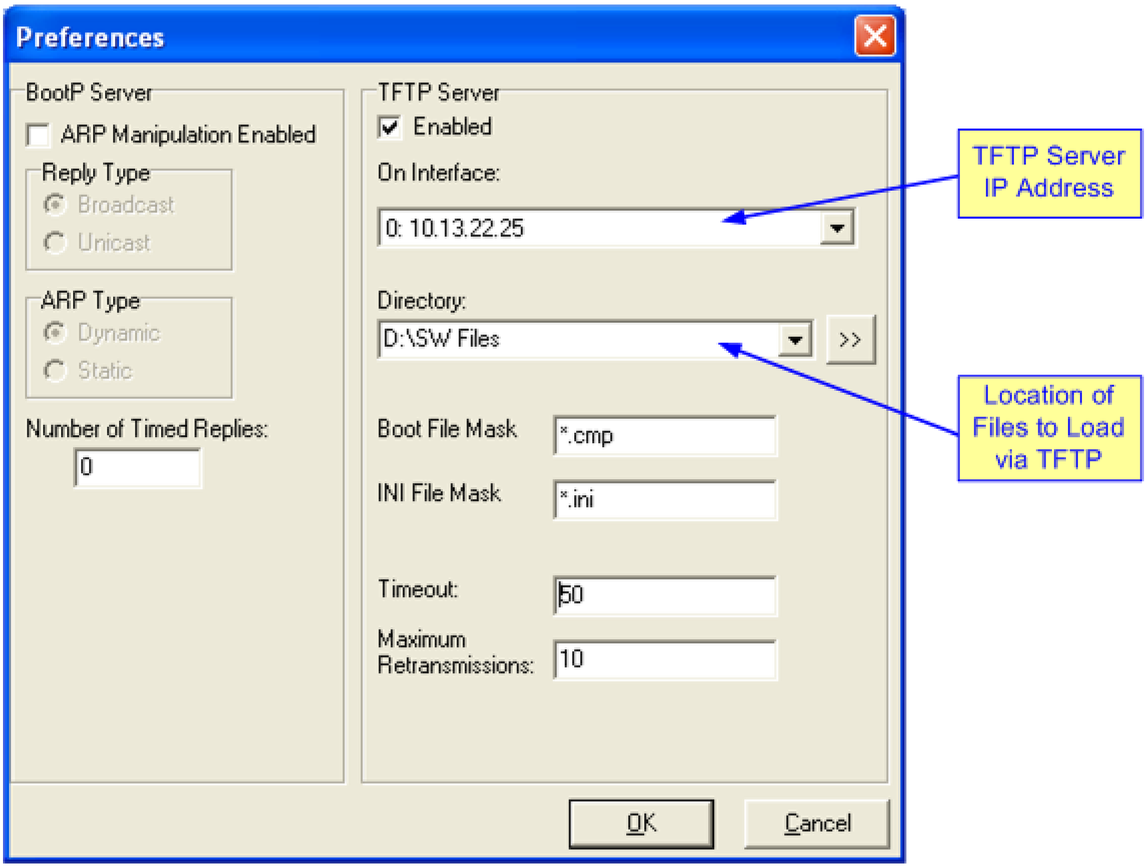
10.1.3 Defining BootP / TFTP Preferences

The first stage is to define BootP / TFTP preferences. Preferences include settings for your BootP server and TFTP server.

➢ To define BootP / TFTP preferences:

1. From the **Edit** menu, choose **Preferences**; the 'Preferences' screen appears:

Figure 10-6: Preferences Screen



Preferences

BootP Server
☐ ARP Manipulation Enabled
Reply Type
● Broadcast
○ Unicast
ARP Type
● Dynamic
○ Static
Number of Timed Replies: 0

TFTP Server
☑ Enabled
On Interface: 0: 10.13.22.25
Directory: D:\SW Files
Boot File Mask: *.cmp
INI File Mask: *.ini
Timeout: 50
Maximum Retransmissions: 10

TFTP Server IP Address

Location of Files to Load via TFTP

2. Define TFTP preferences in the TFTP Server pane:

   - **Enabled:** Select this check box to enable the TFTP functionality of the utility. If you want to use another TFTP application other than the one included with the utility, clear this check box.

- **On Interface:** From the drop-down list, select the network interface available on your PC that you want to use for the TFTP server. (Typically, only one interface is listed.)

Note: The 'On Interface' field is automatically set to the IP address of the PC on which the utility is running.

- **Directory:** This option is enabled only when TFTP is enabled. Specify the folder that contains the files for the TFTP utility to manage (*cmp*, *ini*, Call Progress Tones, etc.).
- **Boot File Mask:** Specify the file extension used by the TFTP utility for the boot file that is included in the BootReply message. This is the file that contains the device's software, i.e., *cmp*.
- **INI File Mask:** Specify the file extension used by the TFTP utility for the configuration file that is included in the BootReply message. This is the file that contains device's configuration parameters, i.e., *ini*.

- **Timeout:** Specifies the number of seconds that the TFTP utility waits before retransmitting TFTP messages. The default value is 30, however, it is recommended to set it to 50 (the more congested your network, the higher you should set this value).
- **Maximum Retransmissions:** Specifies the number of times that the TFTP utility tries to resend messages after timeout. This can be left at the default value of 10 (the more congested your network, the higher you should set this value).

Note: When upgrading the device between major software releases (e.g., from 6.0 to 6.2), it is recommended to set the maximum retransmissions to 20.

3. Define ARP preferences in the BootP Server pane:

   Address Resolution Protocol (ARP) is the method used by all Internet devices to determine the link layer address such as the Ethernet MAC address to route Datagrams to devices that are on the same subnet. When ARP Manipulation is enabled, the BootP/TFTP utility creates an ARP cache entry on your computer when it receives a BootP BootRequest from the device. Your computer uses this information to send messages to the device without using ARP again. This is particularly useful when the device does not yet have an IP address and, therefore, cannot respond to an ARP. Because this feature creates an entry in the computer ARP cache, administrator privileges are required. If the computer is not set to allow administrator privileges, ARP Manipulation cannot be enabled.

   - **ARP Manipulation Enabled:** Enable ARP Manipulation to remotely reset a device that does not yet have a valid IP address. If ARP Manipulation is enabled, the following two option groups are available:
     - **Reply Type:** Reply to a BootRequest can be either **Broadcast** or **Unicast**. The default is **Broadcast** and for the reply to be set to **Unicast**, **ARP Manipulation** must first be enabled. This then enables the BootP / TFTP utility to find the MAC address for the client in the ARP cache so that it can send a message directly to the requesting device. Typically, this setting can be left at **Broadcast**.

- **ARP Type:** The type of entry (**Dynamic** or **Static**) made in the ARP cache on the computer once **ARP Manipulation** is enabled. **Dynamic** entries (default) expire after a period of time, keeping the cache clean so that old entries do not consume computer resources. Static entries do not expire.

4. **Number of Timed Replies:** This is useful for communicating to devices that are located behind a firewall that would block their BootRequest messages from getting through to the computer that is running BootP / TFTP. You can set this value to any whole digit. Once set, BootP / TFTP can send that number of BootReply messages to the destination immediately after you send a remote reset to a device at a valid IP address. This enables the replies to pass through to the device even if the BootRequest is blocked by the firewall. To turn off this feature, set the **Number of Timed Replies** to 0.

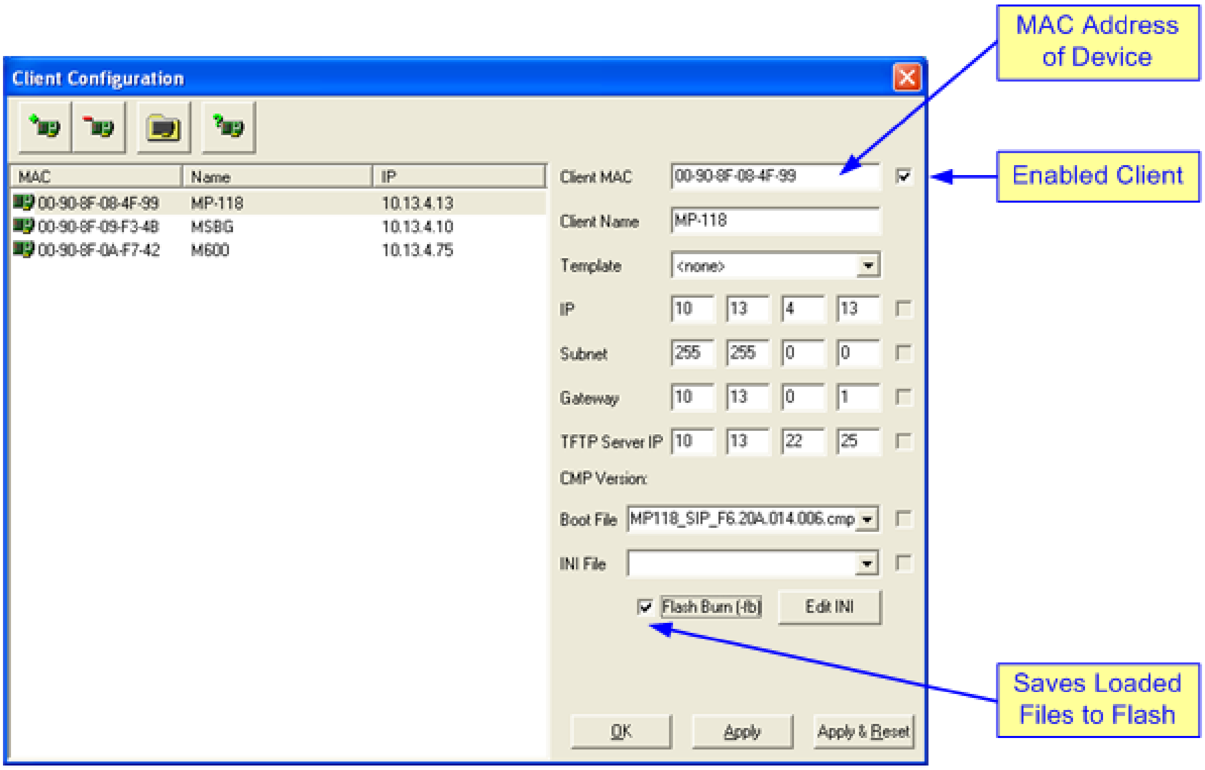
10.1.4 Defining Clients

This section describes how to configure BootP / TFTP clients. The clients are the AudioCodes devices that you want to configure with BootP / TFTP, and are defined by their MAC address.

➢ To add a client:

1. From the **Services** menu, choose **Clients**; the 'Client Configuration' screen appears.

Figure 10-7: Client Configuration Screen



Client Configuration

MAC	Name	IP
00-90-8F-08-4F-99	MP-118	10.13.4.13
00-90-8F-09-F3-4B	MSBG	10.13.4.10
00-90-8F-0A-F7-42	M600	10.13.4.75

Client MAC: 00-90-8F-08-4F-99 ☑

Client Name: MP-118

Template: <none>

IP: 10 13 4 13 □

Subnet: 255 255 0 0 □

Gateway: 10 13 0 1 □

TFTP Server IP: 10 13 22 25 □

CMP Version:

Boot File: MP118_SIP_F6.20A.014.006.cmp □

INI File: □

☑ Flash Burn (-fb)

MAC Address of Device

Enabled Client

Saves Loaded Files to Flash

2. **Click** the **Add New Client** button.

3. Define the client's parameters:

   - **Client MAC**: Enter the Ethernet MAC address of the device. The MAC address of the device is printed on a label located on the device hardware. BootP uses the MAC address to identify the device. Select the check box to the right of this field to enable this client in the BootP (if the client is disabled, no replies are sent to BootP requests).
   - **Client Name:** Enter an optional, descriptive name for the client so that you can easily identify it later.
   - **IP:** Enter the IP address (in dotted-decimal notation) that you want to assign to the device.
   - **Subnet:** Enter the subnet mask (in dotted-decimal notation) that you want to assign to the device.
   - **Gateway:** Enter the IP address of the default gateway used on this subnet that you want to assign to the device.
   - **TFTP Server IP:** Enter the IP address of the TFTP server for transferring software and *ini* files to the device. When creating a new client, this field is populated with the IP address used by the utility. If a different TFTP server utility is used, change the IP address to the IP address used by the other utility.
   - **Boot File:** Specify the file name for the software file (cmp) that is loaded by the TFTP server to the device. The software file is located in the TFTP utility directory, specified in Section 10.1.3. You can also use command-line switches in this field, as described in Section 10.1.5.
   - **INI File:** Select the *ini* file that you want to load to the device. The *ini* file is located in the TFTP utility directory, specified in Section 10.1.3.
   - Select the **Flash Burn** check box (to save the software to the device's non-volatile memory).

4. Click **Apply** to save the new client.

5. Click **OK**; the Client Configuration window closes and the main window appears.

Note: An easy way to create several clients that use similar settings is to create a template and then choose it from the **Template** drop-down list. When you do this, values provided by the template are automatically entered into the parameter fields. To use the template parameters, leave the check boxes corresponding to each parameter selected. To change a parameter to a different value, clear the check box corresponding to the parameter and enter another value. Clicking the check box again restores the template settings. For information on how to create a template, see "Managing Client Templates" on page 194.

- **To delete a client:** Select the client and then click the **Delete Current Client** button.

- **To test that only one BootP utility currently supports a client on the network:**

  Select the client and then click the **Test Selected Client** button. In the Log area of the main window, check that there is no other BootP utility supporting this client MAC (indicated in the Status column as Listed At together with the IP address of that utility). If there is another utility responding to this client, you must remove that client from either this utility or the other one.

10.1.5 Using Command Line Switches

When defining a client (as described in Section 10.1.4), in the 'Boot File'field you can add command-line switches, as described in the procedure below.

➢ **To use a command line switch:**

1. In the field 'Boot File', place your cursor after the *cmp* file name.
2. Press the space bar.
3. Type in the switch you require.

For example:

- ‘ramxxx.*cmp* –fb’ to burn flash memory.
- ‘ramxxx.*cmp* -fb -em 4’ to burn flash memory and for Ethernet Mode 4 (auto-negotiate).

The table below lists and describes the switches that are available:

Table 10-1: Command Line Switch Descriptions

<table>
<tr><th>Switch</th><th colspan="2">Description</th></tr>
<tr><td><b>-fb</b></td><td colspan="2">Burns the <i>cmp</i> file (software file) to the flash memory.
<b>Note:</b> Instead of using this switch, you can simply select the <b>Flash Burn</b> check box.</td></tr>
<tr><td><b>-em #</b></td><td colspan="2">Defines the Ethernet mode:
▪ <b>0</b> = 10Base-T half-duplex (Not applicable to 3000 Series)
▪ <b>1</b> = 10Base-T full-duplex
▪ <b>2</b> = 100Base-TX half-duplex (Not applicable to 3000 Series)
▪ <b>3</b> = 100Base-TX full-duplex
▪ <b>4</b> = auto-negotiate (default)
For detailed information on Ethernet interface configuration, refer to the device's <i>User's Manual</i>.</td></tr>
<tr><td rowspan="3"><b>-br</b></td><td colspan="2">This parameter is used to perform the following:</td></tr>
<tr><td>Defines the number of BootP requests the device sends during startup. The device stops sending BootP requests when either BootP reply is received or number of retries is reached.
▪ <b>1</b> = 1 BootP retry, 1 second
▪ <b>2</b> = 2 BootP retries, 3 seconds
▪ <b>3</b> = 3 BootP retries, 6 seconds
▪ <b>4</b> = 10 BootP retries, 30 seconds
▪ <b>5</b> = 20 BootP retries, 60 seconds
▪ <b>6</b> = 40 BootP retries, 120 seconds
▪ <b>7</b> = 100 BootP retries, 300 seconds
▪ <b>15</b> = BootP retries indefinitely</td><td>Defines the number of DHCP packets the device sends. After all packets are sent, if there's still no reply, the device loads from flash.
▪ <b>1</b> = 4 DHCP packets
▪ <b>2</b> = 5 DHCP packets
▪ <b>3</b> = 6 DHCP packets (default)
▪ <b>4</b> = 7 DHCP packets
▪ <b>5</b> = 8 DHCP packets
▪ <b>6</b> = 9 DHCP packets
▪ <b>7</b> = 10 DHCP packets
▪ <b>15</b> = 18 DHCP packets</td></tr>
<tr><td colspan="2"><b>Note:</b> This switch takes effect only from the next device reset.</td></tr>
</table>

Switch	Description
-bd	Defines the interval between the device's startup and the first BootP/DHCP request that is issued by the device (BootP delays). The switch only takes effect from the next reset of the device. ▪ **1** = 1 second delay (default) ▪ **2** = 10 second delay ▪ **3** = 30 second delay ▪ **4** = 60 second delay ▪ **5** = 120 second delay
-bs	▪ **–bs 1:** enables the Selective BootP mechanism ▪ **–bs 0:** disables the Selective BootP mechanism The Selective BootP mechanism enables the device's integral BootP client to filter unsolicited BootP/DHCP replies (accepts only BootP replies that contain the text 'AUDC' in the Vendor Specific Information field). This option is useful in environments where enterprise BootP/DHCP servers provide undesired responses to the device's BootP requests.

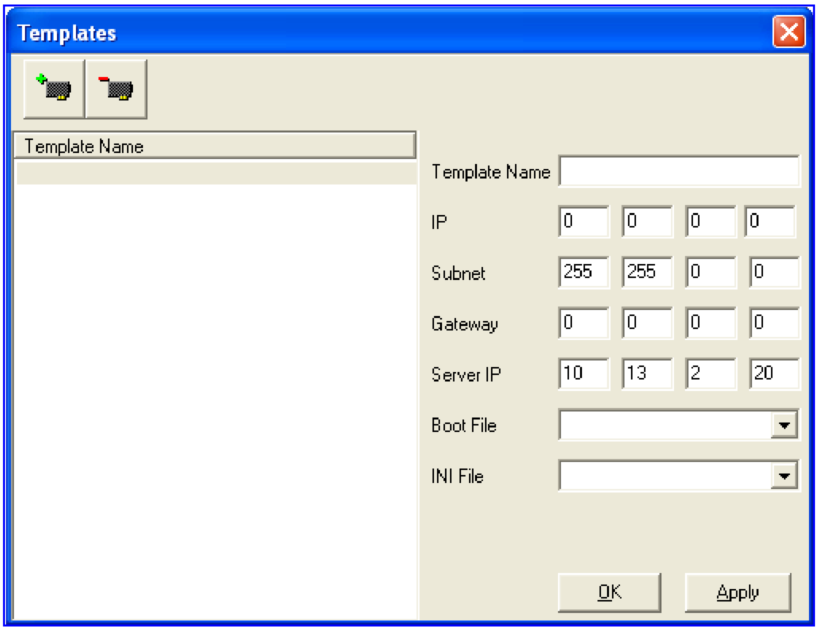
10.1.6 Defining Client Templates

The Templates window (**Services** menu > **Templates**) can be used to add templates to simplify configuration of clients when most of the parameters are identical.

➢ To add a new template:

1. From the Services menu, choose **Templates**; the Templates window appears.

Figure 10-8: Templates Screen



Template Name

IP 0 0 0 0

Subnet 255 255 0 0

Gateway 0 0 0 0

Server IP 10 13 2 20

Boot File

INI File

2. Click the **Add New Template** button.
3. Fill in the required parameter values in the parameter fields.

4. Click **Apply** to save each template.
5. Click **OK** when you are finished adding all your templates.

- To delete a template: select the template, and then click the **Delete Current Template** button.

> **Note:** A template cannot be deleted if it is currently in use.

10.1.7 Resetting Device to Initiate BootP / TFTP

Once you have defined your BootP/TFTP preferences and the client in the utility, you can then initiate the BootP/TFTP process, as described in the procedure below.

> **Notes:**
>
> - If an *ini* file is not specified in the BootP process, the device's current configuration (except for the networking parameters specified in BootP) is retained.
> - To restore the configuration to factory defaults, load an empty *ini* file to the device.

➢ To reset the device to initiate the Boot / TFTP process:

1. Verify that the BootP / TFTP utility is not paused - the pause button must not be selected.
2. Reset the device by doing one of the following:
   - Press the hardware reset pinhole button located on the device and then release.
   - Disconnect and then reconnect power to the device.

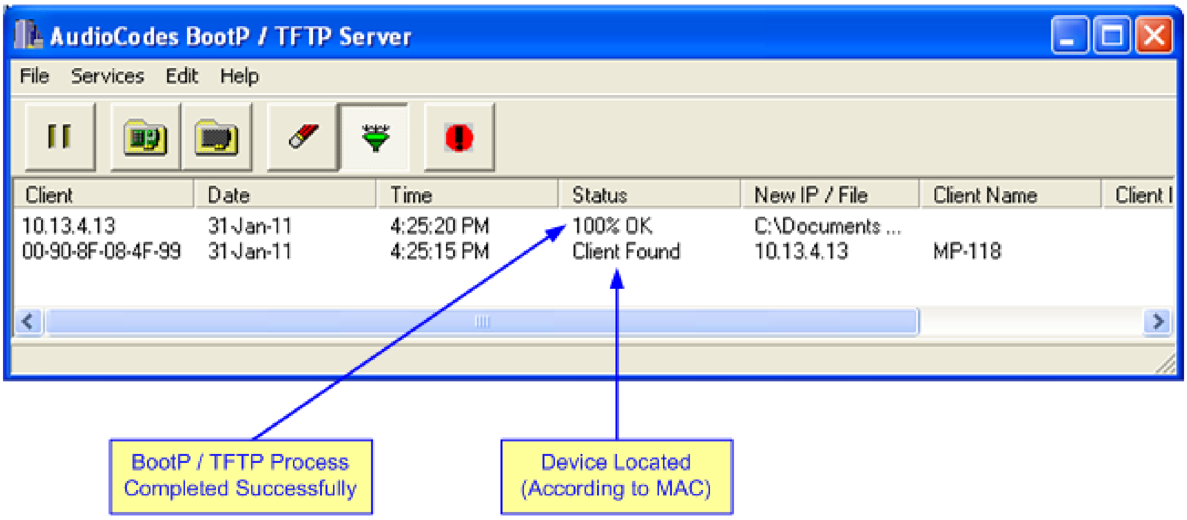
   The BootP server waits for a BootP request from a client that has the specified MAC address. Upon a request, the BootP then assigns the device the specified IP address and then uploads the files to the device.

The main window of the BootP/TFTP utility logs all BootP requests and replies, and TFTP transactions, displaying the progress of the BootP process, as follows:

a. The first indication is that the device (client) with this MAC address was located.

b. The second indication shows the progress of uploading files to the device. When "100% OK", the BootP completed successfully.

Figure 10-9: Main Window Displaying BootP Progress



Client	Date	Time	Status	New IP / File	Client Name	Client I
10.13.4.13	31-Jan-11	4:25:20 PM	100% OK	C:\Documents ...		
00-90-8F-08-4F-99	31-Jan-11	4:25:15 PM	Client Found	10.13.4.13	MP-118	

BootP / TFTP Process Completed Successfully

Device Located (According to MAC)

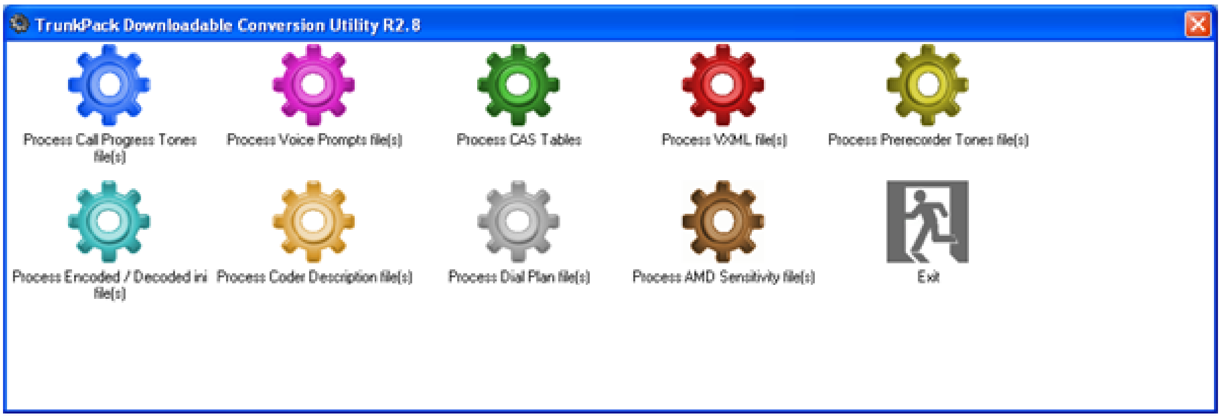
10.2 Downloadable Conversion Utility (DConvert)

The TrunkPack Downloadable Conversion Utility (DConvert) is used to perform the following:

- Create a loadable Call Progress Tones (CPT) file (see "Converting a CPT ini File to a Binary dat File" on page 198)
- Create a loadable Voice Prompts (VP) file from prerecorded voice messages (Applicable only to 3000 Series and 2000 Series devices) (see "Creating a Loadable Voice Prompts File" on page 200)
- Create a loadable CAS protocol table file (Applicable only to Digital PSTN devices) (see "Creating a loadable CAS Protocol Table" on page 202)
- Create Dial Plan file(s) (Applicable only to Digital PSTN devices)
- Encode / decode an *ini* file (see "Encoding / Decoding an ini File" on page 205)
- Create a loadable Prerecorded Tones file (see "Creating a Loadable Prerecorded Tones File" on page 206)
- Creeat a loadable AMD Sensitivity file (see “Creating a Loadable AMD Sensitivity File” on page 209)

The DConvert is run by clicking the file *DConvert.exe*, supplied with your software package.

Figure 10-10: TrunkPack Downloadable Conversion Utility Main Screen


TrunkPack Downloadable Conversion Utility R2.8
Process Call Progress Tones file(s)
Process Voice Prompts file(s)
Process CAS Tables
Process VXML file(s)
Process Prerecorder Tones file(s)
Process Encoded / Decoded ini file(s)
Process Coder Description file(s)
Process Dial Plan file(s)
Process AMD Sensitivity file(s)
Exit

Note: The 'Process VXML file(s)' and 'Process Coder Description files(s)' options are not applicable to SIP devices.

10.2.1 Converting a CPT ini File to a Binary dat File

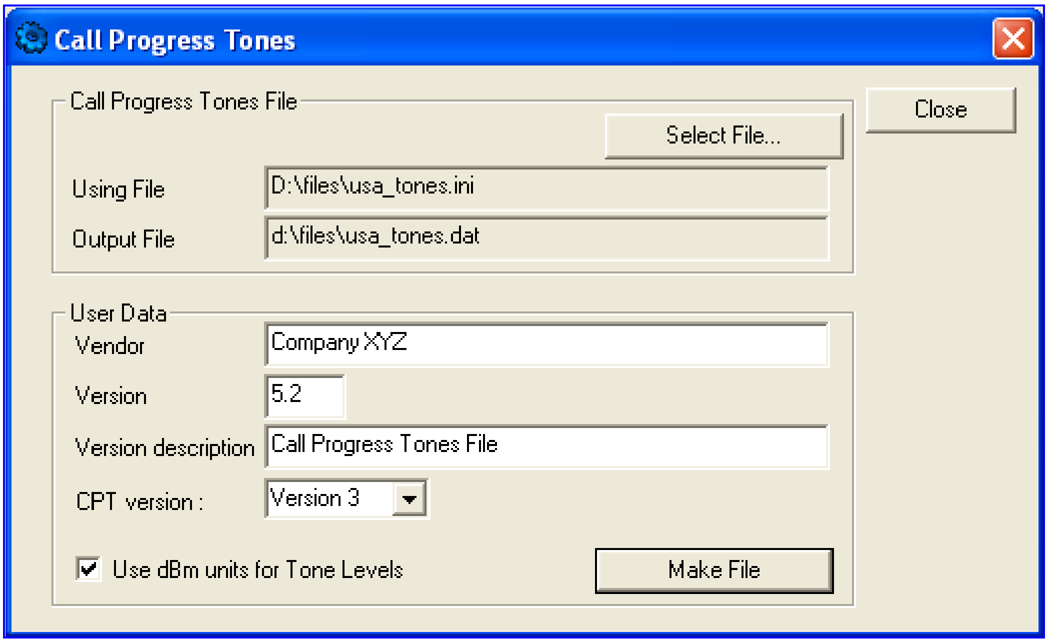
The procedure below describes how to convert a Call Progress Tones (CPT) *ini* file to a binary *.dat file, using DConvert. For detailed information on creating a CPT *ini* file, see Configuring the Call Progress Tones and Distinctive Ringing File in the device's *User's Manual*.

➢ **To convert a CPT *ini* file to a binary *dat* file:**

1. Start DConvert; the main window opens (shown in "TrunkPack Downloadable Conversion Utility" on page 195).

2. Click the **Process Call Progress Tones File(s)** ⚙ button; the 'Call Progress Tones' dialog box opens, shown in the figure below.

Figure 10-11: Call Progress Tones Dialog Box



Using File: D:\files\usa_tones.ini

Output File: d:\files\usa_tones.dat

Vendor: Company XYZ

Version: 5.2

Version description: Call Progress Tones File

CPT version : Version 3

☑ Use dBm units for Tone Levels

3. Under the 'Call Progress Tones File' group, click the **Select File** button.

4. Navigate to the folder that contains the CPT *ini* file that you want to convert.

5. Select the *ini* file, and then click the **Open** button; the name and path of both the *ini* file and the (output) *dat* file appears in the fields below the **Select File** button.

6. Under the 'User Data' group, enter the perform the following:
   a. In the 'Vendor' field, enter the vendor's name (maximum length is 256 characters).
   b. In the 'Version' field, enter the version number. The format is composed of two integers separated by a period '.' (e.g., 1.2, 23.4, 5.22)/
   c. In the 'Version Description' field, enter a brief description of this file. The maximum length is 256 characters.

7. The default value of the 'CPT Version' drop-down list is Version 3. Perform one of the following:

   - If the software version you are using is prior to version 4.4, select Version 1 (to maintain backward compatibility).
   - If the software version you are using is 4.4, select Version 2.
   - Otherwise, leave the value at its default.

8. Select the 'Use dBm units for Tone Levels' check box. Note that the levels of the call progress tones (in the CPT file) must be in -dBm units.

9. Click the **Make File** button; the file is created and a message box is displayed when successfully complete.

10. Close the application.

10.2.2 Creating a Loadable Voice Prompts File

The procedure below describes how to create a loadable Voice Prompts file, using DConvert. For detailed information on the Voice Prompts file, refer to Voice Prompts File in the device's *User's Manual*.

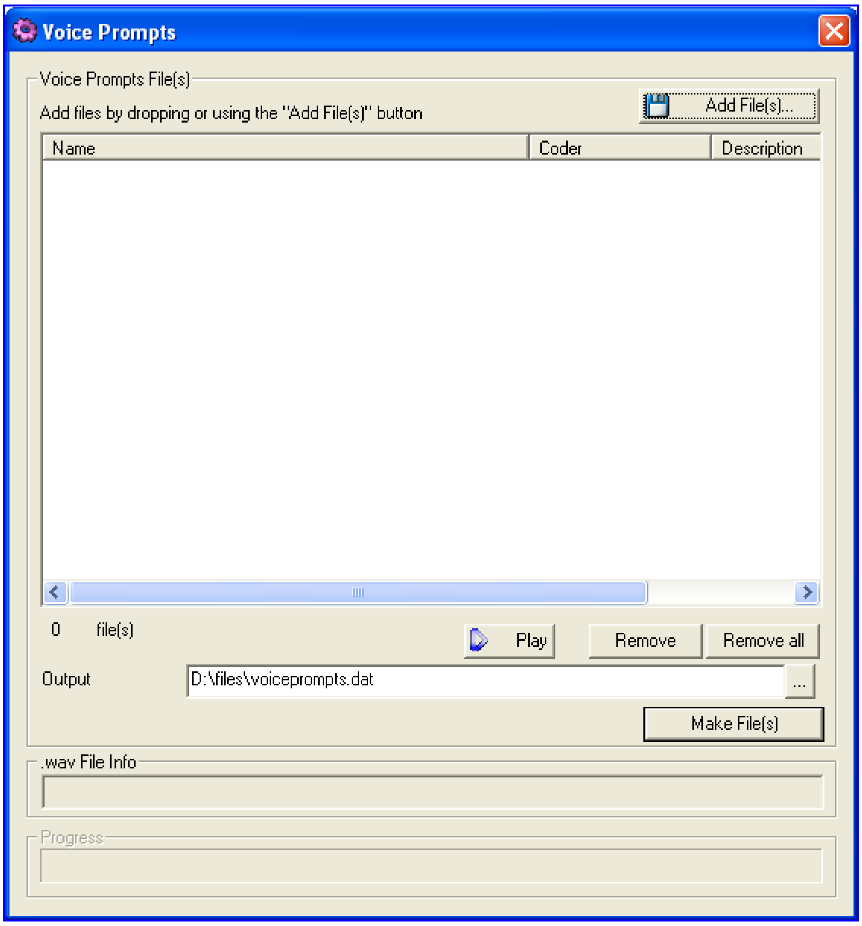
Note: This subsection is applicable only to 3000 Series and 2000 Series devices.

➢ **To create a loadable Voice Prompts *dat* file from your voice recording files:**

1. Start DConvert; the main window appears (shown in "TrunkPack Downloadable Conversion Utility" on page 195).

2. Click the **Process Voice Prompts File(s)** button; the 'Voice Prompts' dialog box opens.

Figure 10-12: Voice Prompts Screen



Output: D:\files\voiceprompts.dat

3. To add the prerecorded voice files to the 'Voice Prompts' screen, perform one of the following:
   - Select the files and drag them into the 'Voice Prompts' screen.
   - Click the **Add File(s)** button; the 'Select Files' screen opens. Select the required Voice Prompt files, and then click the **Add** button. Close the 'Select Files' screen.
4. Arrange the files according to your requirements by dragging and dropping them from one location in the list to another. Note that the order of the files determines their assigned Voice Prompt ID.

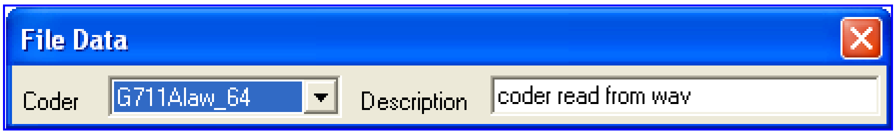
Tips:

- Use the **Play** button to listen to the *wav* files.
- Use the **Remove** and **Remove all** buttons to delete files from the list.

5. For each of the raw files, select a coder that corresponds to the coder in which it was originally recorded, by completing the following steps:
   a. Double-click or right-click the required file(s); the 'File Data' window (shown in the figure below) appears.
   b. From the 'Coder' drop-down list, select the required coder type.
   c. In the 'Description' field, enter additional identifying information.
   d. Close the 'File Data' window.

   Note: For *wav* files, a coder is automatically selected from the wav file's header.

Figure 10-13: File Data Window



File Data

Coder: G711Alaw_64

Description: coder read from wav

6. In the 'Output' field, specify the directory to which the Voice Prompts file is generated, followed by the name of the Voice Prompts file (the default name is voiceprompts.dat).
7. Click the **Make File(s)** button; the Voice Prompts loadable file is produced.

10.2.3 Creating a Loadable CAS Protocol Table File

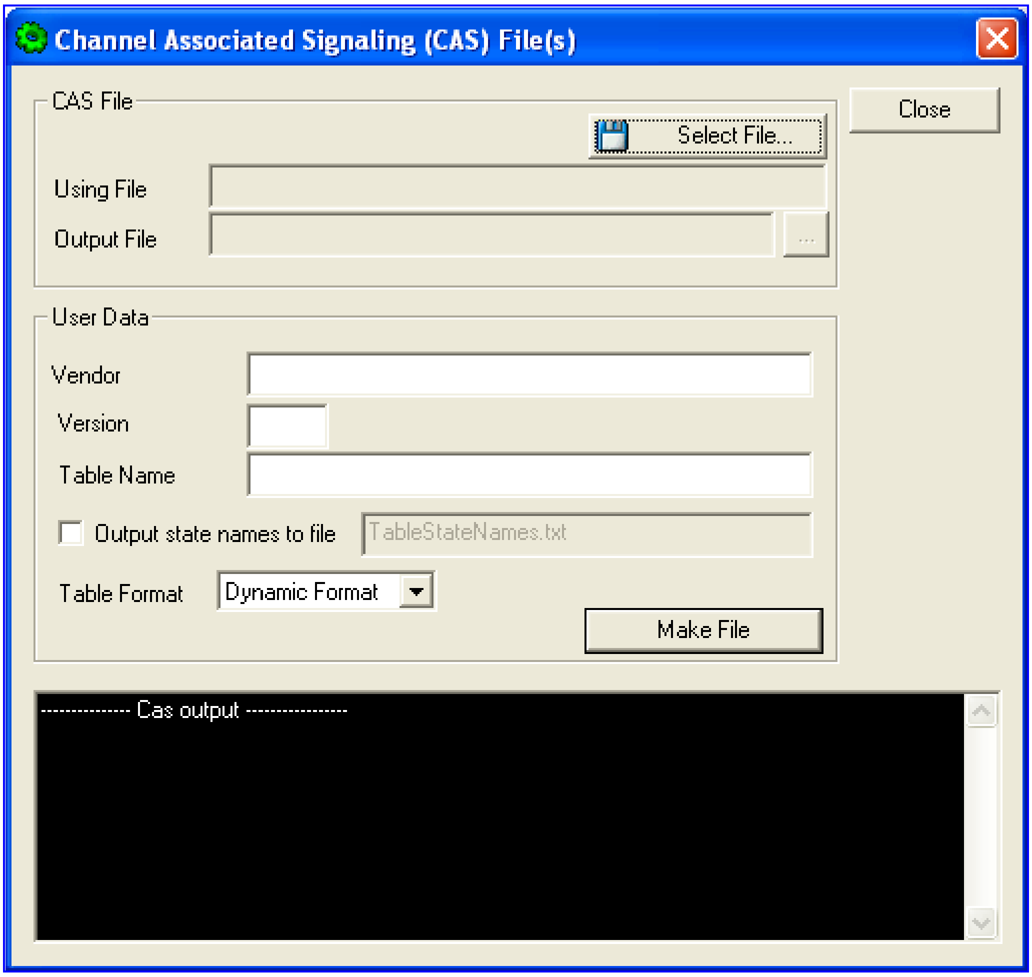
The procedure below describes how to create a loadable CAS Protocol Table file, using DConvert.

> **Note:** This subsection is applicable only to Digital PSTN devices.

➢ To create a loadable CAS protocol table file:

1. Create the CAS protocol files (*xxx.txt* and *UserProt_defines_xxx.h*).
2. Copy the files generated in the previous step to the same directory in which DConvert is located. Ensure that the files *CASSetup.h* and *cpp.exe* are also located in the same directory.
3. Start DConvert; the main window opens (shown in "TrunkPack Downloadable Conversion Utility" on page 195).
4. Click **Process CAS Tables** ⚙ button; the Channel Associated Signaling (CAS) screen opens, shown in the figure below.

Figure 10-14: Call Associated Signaling (CAS) Screen



□ Output state names to file: TableStateNames.txt

Table Format: Dynamic Format

```
-------------- Cas output ----------------
```

5. Under the 'CAS File' group, click **Select File**, navigate to the folder in which the file is located, and then select the *txt* file you want converted; the 'Output File' field displays the file name and path, but with a *dat* extension. The table's name is also automatically designated.

6. Under the 'User Data' group, perform the following:

   a. In the 'Vendor' field, enter the vendor's name (maximum of 32 characters).

   b. In the 'Version' field, enter the version number. The value must be in the following format: [number] [single period '.'] [number] (e.g., 1.2, 23.4, 5.22)

7. In the 'Table Name' field, modify the name according to your requirements.

8. To create a file (for troubleshooting purposes) that contains the name of the States and their actual values, select the 'Output state names to file' check box; the default file name *TableStateNames.txt* appears in the adjacent field (you can modify the name of the file). The generated file is to be located in the same directory as DConvert.

9. From the 'Table Format' drop-down list, select the format you want to use:

   - Old Format: supported by all versions. Many CAS features are not supported in this format.

   - New Format: supported from 4.2 and later. From 5.2 and later a few new features are not supported by this format.

   - Dynamic Format: supported from 5.2 and later. Some 5.2 features are only supported by this format. The size of the file with dynamic format is significantly lower that other formats.

10. Click **Make File**; the *dat* file is generated and saved in the directory specified in the 'Output File' field. A message box informing you that the operation was successful indicates that the process is completed. In the pane at the bottom of the Call Assisted Signaling (CAS) Files(s) screen, the CAS output log box displays the log generated by the process. It can be copied as needed. The information in it isn't retained after the screen is closed.

10.2.4 Creating a Dial Plan File

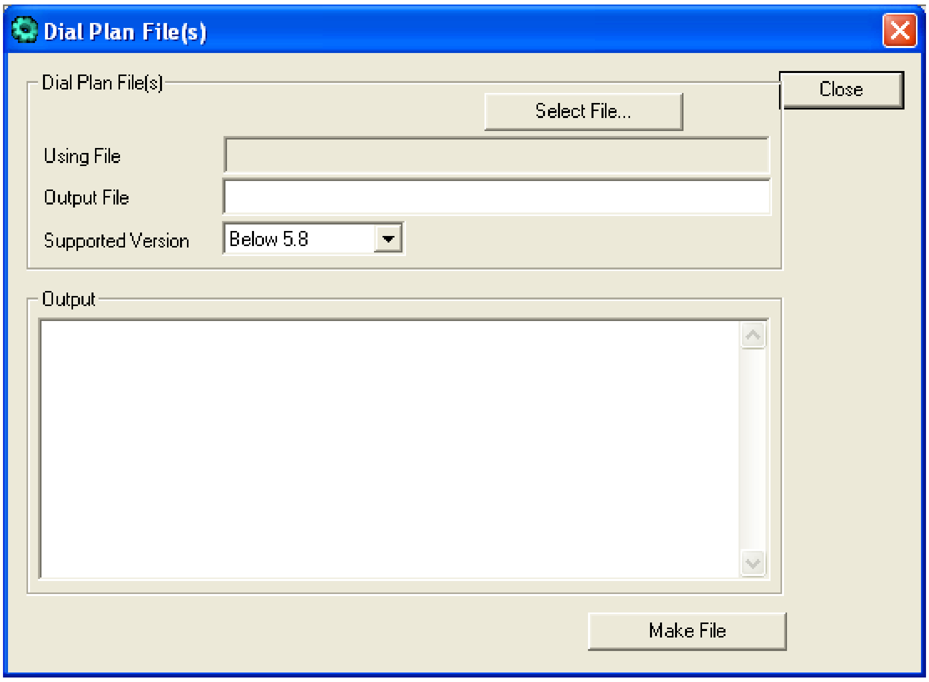
The procedure below describes how to create a Dial Plan file, using DConvert.

Note: This subsection is applicable only to Digital PSTN devices.

➢ **To create a Dial Plan file:**

1. Construct a Dial Plan text file according to the instructions in Dial Plan File in the device's *User's Manual*.
2. Start DConvert; the main window appears.
3. Click the **Process Dial Plan File(s)** ⚙ button; the 'Dial Plan File(s)' window appears.

Figure 10-15: Dial Plan Screen



4. Click the **Select File** button, navigate to the desired folder, and then select the file to be converted; the selected file name (but with the *.dat* extension) and path is displayed in the 'Output File' field. The output file name may be altered.
5. Click the **Make File** button. The *.dat* file is generated and saved in the same directory as shown in the 'Output File' field. A message box informing you that the operation was successful indicates that the process has been completed.
6. On the bottom of the 'Coders' window, the 'Output' log box displays the log generated by the process. It may be copied as needed. This information is not retained after the window is closed.

Note: The process verifies the input file for validity. Invalid data causes an error and aborts the process. In such a case, the log box contains further information.

10.2.5 Encoding / Decoding an ini File

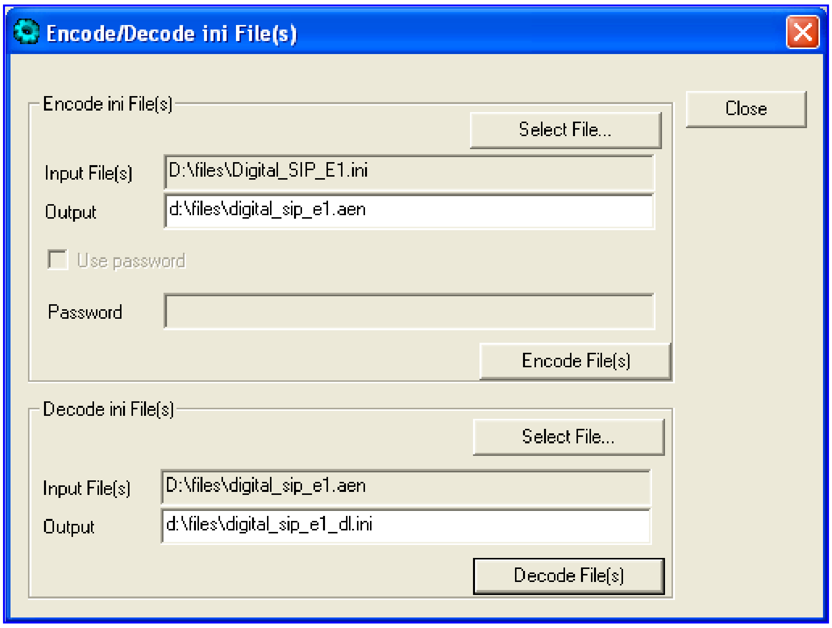
The procedure below describes how to encode and decode an *ini* file, using DConvert. For detailed information on secured *ini* file, refer to Secured ini File in the device's *User's Manual*.

➢ **To encode an *ini* file:**

1. Start DConvert; the main window opens (shown in in "TrunkPack Downloadable Conversion Utility" on page 195).

2. Click the **Process Encoded/Decoded ini file(s)** ⚙ button; the 'Encode/Decode *ini* File(s)' screen, shown below, opens.

Figure 10-16: Encode / Decode ini File(s) Screen



Encode/Decode ini File(s)

Encode ini File(s)

Select File...

Input File(s): D:\files\Digital_SIP_E1.ini

Output: d:\files\digital_sip_e1.aen

□ Use password

Password:

Encode File(s)

Decode ini File(s)

Select File...

Input File(s): D:\files\digital_sip_e1.aen

Output: d:\files\digital_sip_e1_dl.ini

Decode File(s)

Close

3. Under the 'Encode *ini* File(s)' group, click the **Select File** button.

4. Navigate to the folder that contains the *ini* file you want to encode.

5. Select the *ini* file, and then click the **Open** button; the name and path of both the *ini* file and the output encoded file appear in the fields under the **Select File** button. Note that the name and extension of the output file can be modified.

6. Click the **Encode File(s)** button; an encoded *ini* file with the name and extension you specified is created.

➢ **To decode an encoded *ini* file:**

1. Under the 'Decode *ini* File(s)' group, click the **Select File** button.

2. Navigate to the folder that contains the file you want to decode.

3. Click the file and click the **Open** button; the name and path of both the encode *ini* file and the output decoded file appear in the fields under the **Select File** button. Note that the name of the output file can be modified.

4. Click the **Decode File(s)** button; a decoded *ini* file with the name you specified is created.

Note: The decoding process verifies the input file for validity. Any change made to the encoded file causes an error and the decoding process is aborted.

10.2.6 Creating a Loadable Prerecorded Tones File

The procedure below describes how to create a loadable Prerecorded Tones (PRT) file, using DConvert. For detailed information on PRT files, refer to Prerecorded Tones (PRT) File in the device's *User's Manual*.

Notes:

- It is highly recommended to avoid using the Linear PCM coder.
- PRT is not applicable to Mediant 800 series, Mediant 4000, and Mediant Software E-SBC.

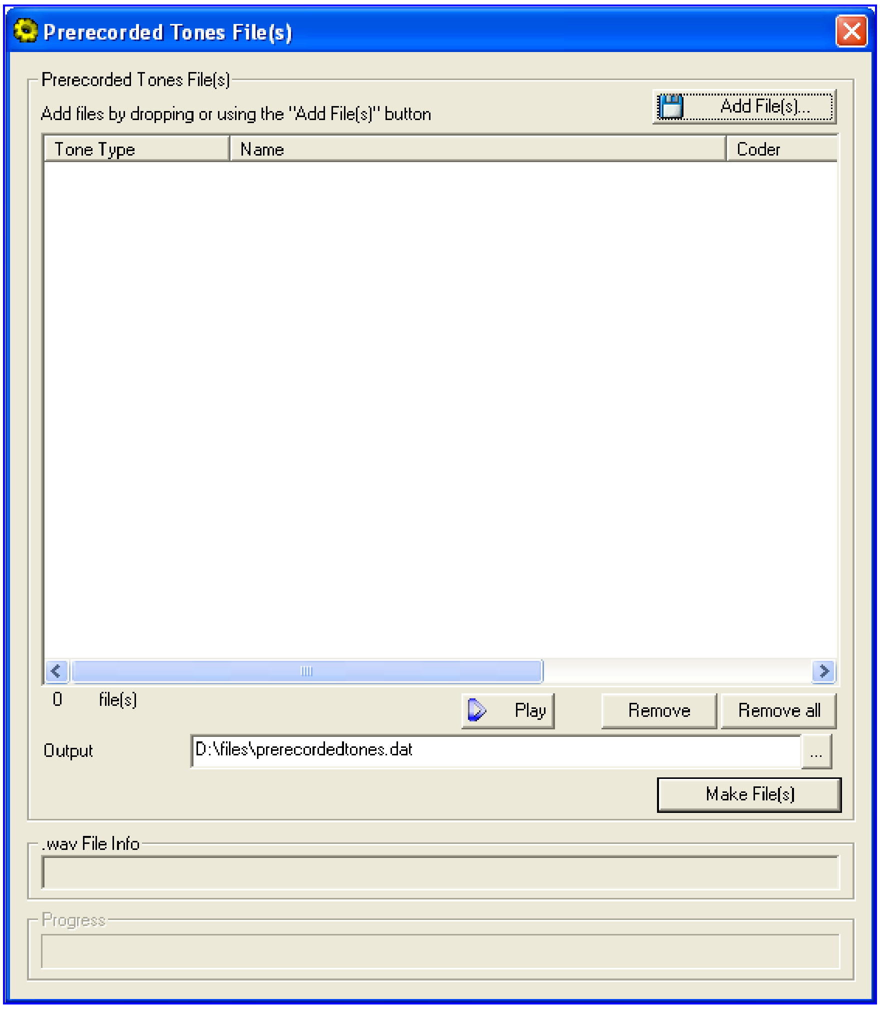
➢ To create a loadable PRT *dat* file from your raw data files:

1. Prepare the PRT files that you want to combine into a single *dat* file using standard recording utilities.

2. Start DConvert; the main window opens (shown in "TrunkPack Downloadable Conversion Utility" on page 195).

3. Click the **Process Prerecorder Tones File(s)** button; the Prerecorded Tones File(s) screen opens.

Figure 10-17: Prerecorded Tones Screen



Add files by dropping or using the "Add File(s)" button

Output D:\files\prerecordedtones.dat

4. To add the PRT files (that you created in Step 1), perform one of the following:

- Select the files and drag them into the 'Prerecorded Tones File(s)' screen.
- Click the **Add File(s)** button; the 'Select Files' screen opens. Select the required PRT files, and then click the **Add** button. Close the 'Select Files' screen.

5. For each raw data file, define a tone type, a coder, and the default duration, by completing the following steps:

   a. Double-click or right-click the required file; the 'File Data' window (shown in the figure below) appears.

   b. From the 'Type' drop-down list, select the tone type with which this raw data file is associated.

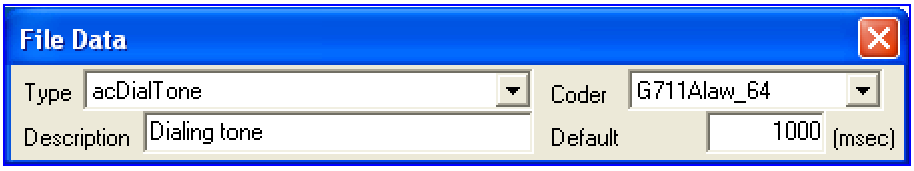
   c. From the 'Coder' drop-down list, select the coder that corresponds to the coder with which this raw data file was originally recorded.

d. In the 'Description' field, enter brief identifying information (optional).

e. In the 'Default' field, enter the default duration this raw data file is repeatedly played.

f. Close the 'File Data' window (press the **Esc** key to cancel your changes); you are returned to the 'Prerecorded Tones File(s)' screen.

Figure 10-18: File Data Window



File Data

Type: acDialTone

Coder: G711Alaw_64

Description: Dialing tone

Default: 1000 (msec)

6. In the 'Output' field, specify the output directory in which the PRT file is generated, followed by the name of the PRT file (the default name is *prerecordedtones.dat*). Alternatively, use the **Browse** button to select a different output file, navigate to the desired file, and then select it; the selected file name and its path appear in the 'Output' field.

7. Click the **Make File(s)** button; the progress bar at the bottom of the window is activated. The *dat* file is generated and saved in the directory specified in the 'Output' field. A message box informing you that the operation was successful indicates that the process is completed.

10.2.7 Creating a Loadable AMD Sensitivity File

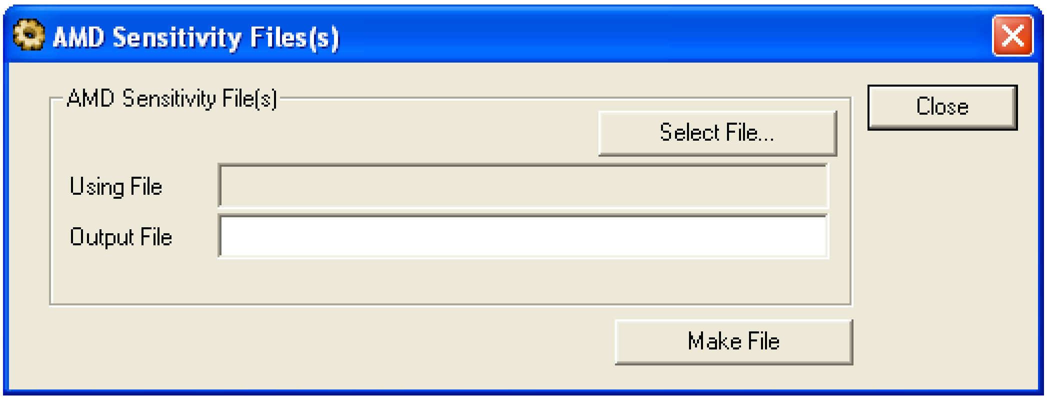
The procedure below describes how to create a loadable AMD Sensitivity file (*.dat), using DConvert. For detailed information on this file, refer to the device's *User's Manual*.

➢ To convert an AMD Sensitivity *.xml file to a binary *dat* file:

1. Start DConvert; the main window opens (shown in "TrunkPack Downloadable Conversion Utility" on page 195).

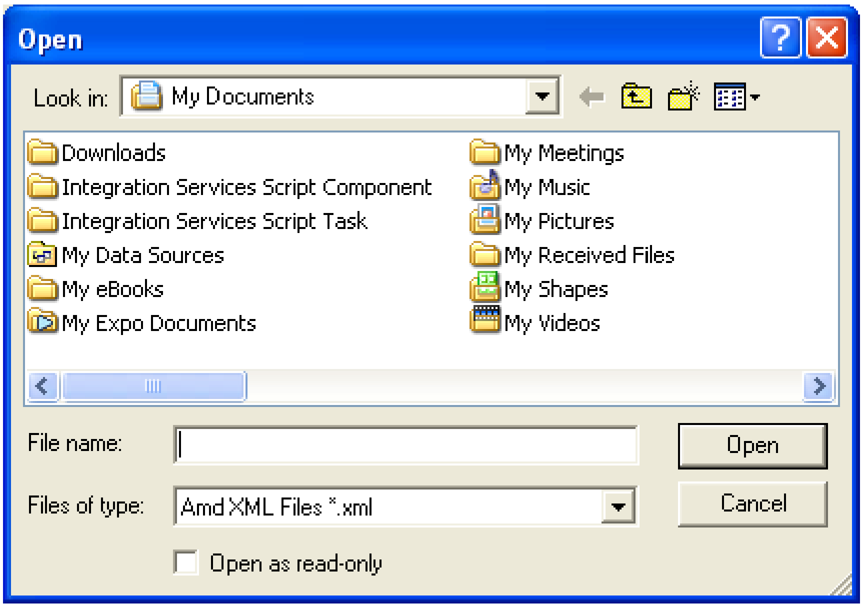
2. Click the **Process AMD Sensitivity File(s)** ⚙ button; the ‘AMD Sensitivity File(s)’ dialog box opens, shown in the figure below.

Figure 10-19: AMD Sensitivity File(s) Dialog Box



3. Click the **Select File** button and then navigate and select the AMD file in XML format.

Figure 10-20: Select File



4. Click the **Make File** button to convert it to a *.dat file.

10.3 Call Progress Tones Wizard

Note: This subsection is applicable only to Analog devices.

The Call Progress Tones Wizard (CPTWizard) is an application designed to facilitate the provisioning of an FXO device by recording and analyzing Call Progress Tones (CPT) generated by any PBX or telephone network. The CPTWizard creates a basic CPT *ini* file and dat files, providing a good starting point when configuring an FXO device. The *ini* file contains definitions for all relevant CPT; the dat file (which can also be created using DConvert -- "Converting a CPT ini File to a Binary dat File" on page 198) is in a format that is suitable for downloading to the device.

To use this wizard, an FXO device connected to your PBX with two physical phone lines is required. This device must be configured with factory-default settings and mustn't be used for phone calls during the operation of the wizard.

Note: You must use the CPTWizard version that corresponds to the device's software version.

10.3.1 Installation

The CPTWizard can be installed on any PC running Windows 2000 or Windows XP. Windows-compliant networking and audio peripherals are required for full functionality. To install the CPTWizard, copy the files from the supplied installation kit to any folder on your PC. No further setup is required (approximately 5 MB of hard disk space is required).

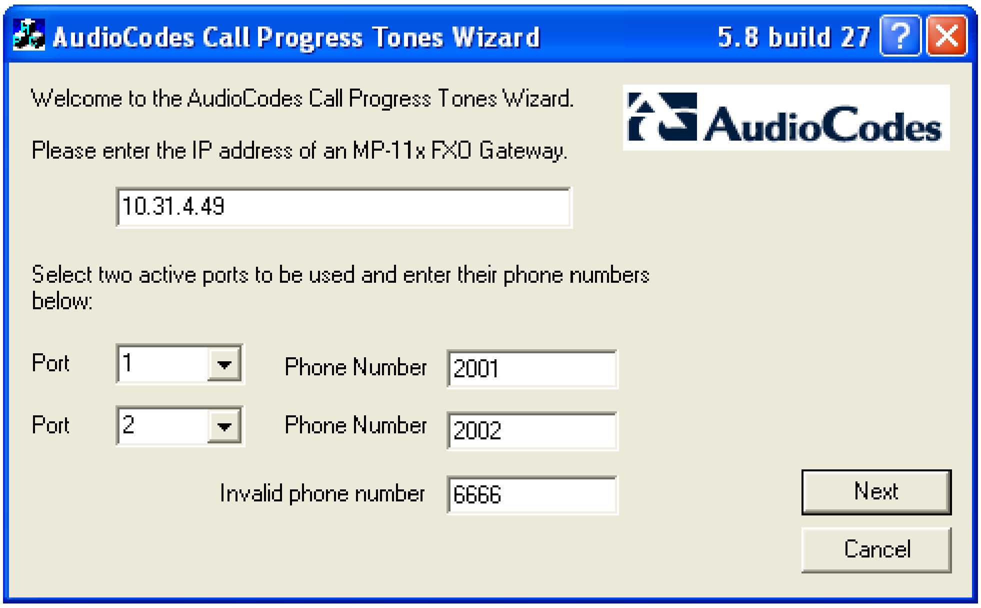
10.3.2 Initial Settings

The procedure below describes how to start the CPTWizard.

➢ To start the CPTWizard:

1. Run the *CPTWizard.exe* file; the wizard's initial settings screen is displayed:

Figure 10-21: Initial Settings Screen



AudioCodes Call Progress Tones Wizard 5.8 build 27

Welcome to the AudioCodes Call Progress Tones Wizard.

Please enter the IP address of an MP-11x FXO Gateway.

10.31.4.49

Select two active ports to be used and enter their phone numbers below:

Port 1 Phone Number 2001

Port 2 Phone Number 2002

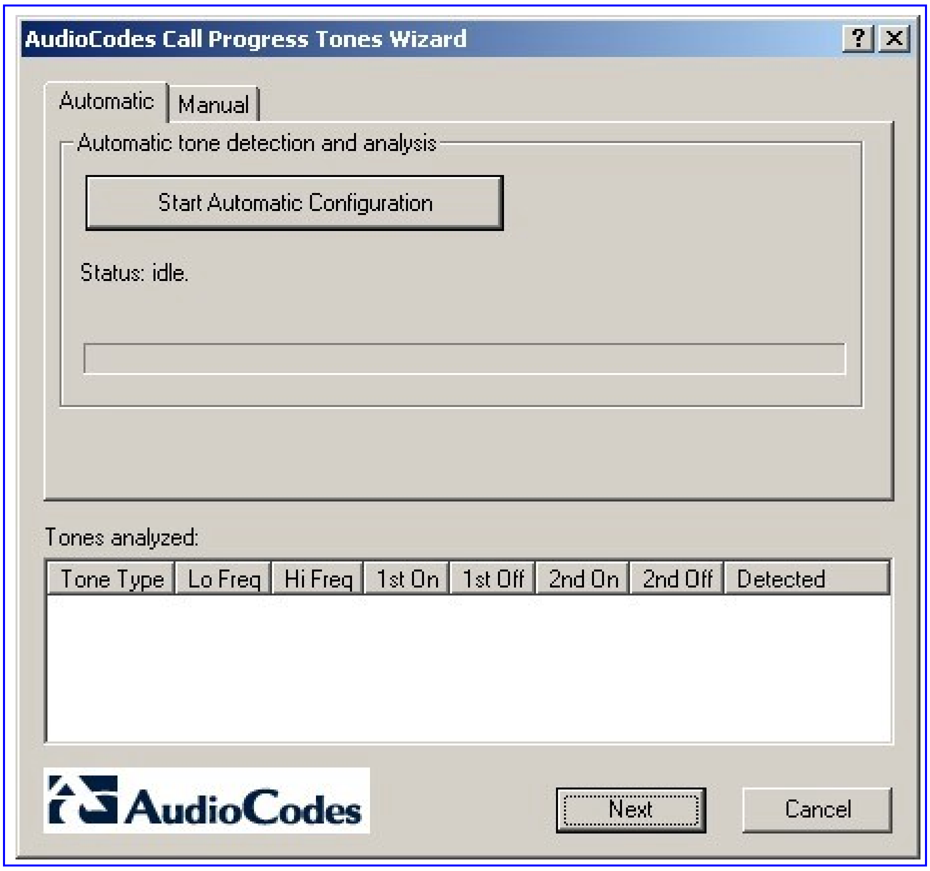
Invalid phone number 6666

2. Enter the IP address of the FXO device.
3. Select the device's ports that are connected to your PBX, and specify the phone number of each extension.
4. In the 'Invalid phone number' field, enter a number that generates a 'fast busy' tone when dialed. Usually any incorrect phone number should cause a 'fast busy' tone.
5. Click **Next**.

10.3.3 Recording Screen - Automatic Mode

Once the connection between the CPTWizard and the FXO device is established, the recording screen is displayed:

Figure 10-22: Recording Screen - Automatic Mode



➢ To start recording in automatic mode:

1. Click the **Start Automatic Configuration** button; the wizard starts the following Call Progress Tones detection sequence (the operation takes approximately 60 seconds to complete):

   a. Sets port 1 to offhook, and then listens to the dial tone.

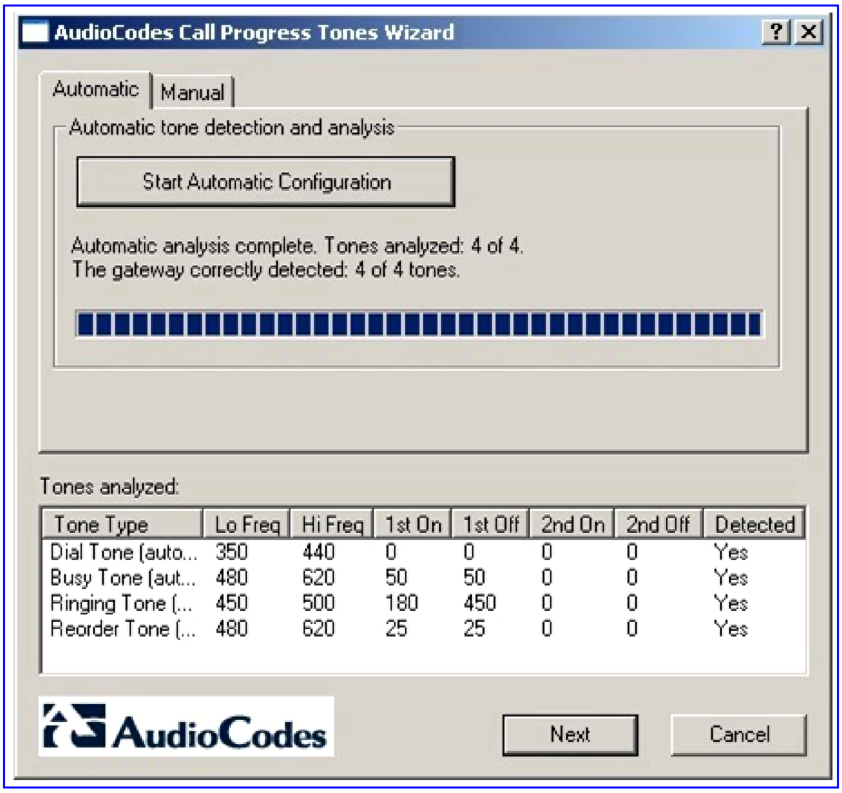
   b. Sets port 1 and port 2 to offhook, dials the number of port 2, and then listens to the busy tone.

   c. Sets port 1 to offhook, dials the number of port 2, and then listens to the Ringback tone.

   d. Sets port 1 to offhook, dials an invalid number, and then listens to the reorder tone.

2. The wizard then analyzes the recorded Call Progress Tones and displays a message specifying the tones that were detected (by the device) and analyzed (by the wizard) correctly. At the end of a successful detection operation, the detected Call Progress Tones are displayed in the Tones Analyzed pane, as shown in the figure below:

Figure 10-23: Recording Screen after Automatic Detection


AudioCodes Call Progress Tones Wizard
Automatic
Manual
Automatic tone detection and analysis
Start Automatic Configuration
AudioCodes
Next
Cancel

Automatic analysis complete. Tones analyzed: 4 of 4.
The gateway correctly detected: 4 of 4 tones.

Tones analyzed:

Tone Type	Lo Freq	Hi Freq	1st On	1st Off	2nd On	2nd Off	Detected
Dial Tone (auto...	350	440	0	0	0	0	Yes
Busy Tone (aut...	480	620	50	50	0	0	Yes
Ringing Tone (...	450	500	180	450	0	0	Yes
Reorder Tone (...	480	620	25	25	0	0	Yes

3. All four Call Progress Tones are saved (as standard A-law PCM at 8000 bits per sample) in the same directory as the *CPTWizard.exe* file is located, with the following names:

   - *cpt_recorded_dialtone.pcm*
   - *cpt_recorded_busytone.pcm*
   - *cpt_recorded_ringtone.pcm*
   - *cpt_recorded_invalidtone.pcm*

4. At this stage, you can either click **Next** to generate a Call Progress Tones *ini* and *dat* files and terminate the wizard, or continue to manual recording mode.

Notes:

- If the device is configured correctly (with a Call Progress Tones *dat* file loaded to the device), all four Call Progress Tones are detected by the device. By noting whether the device detects the tones or not, you can determine how well the Call Progress Tones *dat* file matches your PBX. During the first run of the CPTWizard, it is likely that the device does not detect any tones.
- Some tones cannot be detected by the FXO device (such as 3-frequency tones and complex cadences). CPTWizard is therefore, limited to detecting only those tones that can be detected on the FXO device.

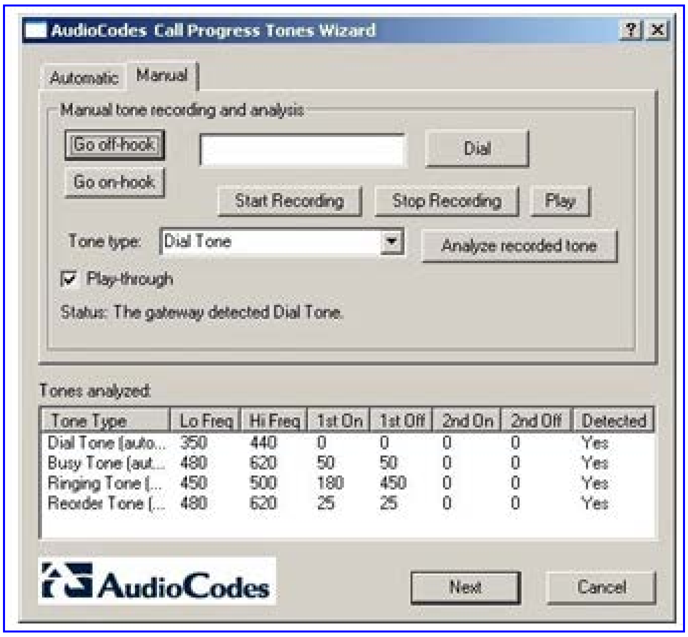
10.3.4 Recording Screen - Manual Mode

In manual mode, you can record and analyze tones included in the Call Progress Tones *ini* and dat files in addition to the tones analyzed when in automatic mode.

➢ To start recording in manual mode:

1. In the recording screen, click the **Manual** tab; the 'Manual Tone Recording' pane is displayed.

Figure 10-24: Recording Screen - Manual Mode



Tone type: Dial Tone

☑ Play-through

Status: The gateway detected Dial Tone.

Tones analyzed:

Tone Type	Lo Freq	Hi Freq	1st On	1st Off	2nd On	2nd Off	Detected
Dial Tone (auto...	350	440	0	0	0	0	Yes
Busy Tone (aut...	480	620	50	50	0	0	Yes
Ringing Tone (...	450	500	180	450	0	0	Yes
Reorder Tone (...	480	620	25	25	0	0	Yes

2. Select the **Play-through** check box to hear the tones through your PC speakers.
3. Click the **Go off-hook** button, enter a number to dial in the 'Dial' field, and then click the **Dial** button.
4. When you're ready to record, click the **Start Recording** button.
5. When the desired tone is complete, click **Stop Recording**. (The recorded tone is saved as 'cpt_manual_tone.pcm'.)

Note: Due to some PC audio hardware limitations, you may hear 'clicks' in play-through mode. You can ignore these clicks.

6. From the 'Tone type' drop-down list, select the tone type, and then click **Analyze recorded tone**; the analyzed tone is added to the 'Tones analyzed' list at the bottom of the screen. It is possible to record and analyze several different tones for the same tone type (e.g., different types of 'busy' signal).
7. Repeat the process for more tones, as necessary.
8. When you're finished adding tones to the list, click **Next** to generate a Call Progress Tones *ini* and dat files and terminate the wizard.

10.3.5 Call Progress Tones ini and dat Files

Once the wizard completes the Call Progress Tone detection, a text file named *call_progress_tones.ini* and a binary file named *call_progress_tones.dat* are created in the same directory in which the *CPTWizard.exe* file is located. The latter is ready for download to the device and it contains the same output which the DConvert utility would produce when processing the *ini* file.

The *ini* file contains the following information:

- Information on each tone that was recorded and analyzed by the wizard. This information includes frequencies and cadence (on/off) times, which is required when converting the *ini* file to dat.

  Below shows an example of an *ini* file with Call Progress Tone properties:

```
[CALL PROGRESS TONE #1]
Tone Type=1
Low Freq [Hz]=350
High Freq [Hz]=440
Low Freq Level [-dBm]=0
High Freq Level [-dBm]=0
First Signal On Time [10msec]=0
First Signal Off Time [10msec]=0
Second Signal On Time [10msec]=0
Second Signal Off Time [10msec]=0
```

- Information relating to possible matches of *each* tone with the CPTWizard's internal database of common tones. This information is specified as comments in the file and is ignored when converting the *ini* file to a dat file.

  Below shows an example of a file with Call Progress Tone database matches:

```
# Recorded tone: Busy Tone (automatic configuration)
## Matches: PBX name=ITU Anguilla, Tone name=Busy tone
## Matches: PBX name=ITU Antigua and Barbuda, Tone name=Busy tone
## Matches: PBX name=ITU Barbados, Tone name=Busy tone
## Matches: PBX name=ITU Bermuda, Tone name=Busy tone
## Matches: PBX name=ITU British Virgin Islan, Tone name=Busy tone
## Matches: PBX name=ITU Canada, Tone name=Busy tone
## Matches: PBX name=ITU Dominica (Commonweal, Tone name=Busy tone
## Matches: PBX name=ITU Hongkong, China, Tone name=Busy tone
## Matches: PBX name=ITU Jamaica, Tone name=Busy tone
## Matches: PBX name=ITU Korea (Republic of), Tone name=Busy tone
## Matches: PBX name=ITU Montserrat, Tone name=Busy tone
```

- Information relating to matches of *all* tones recorded with the CPTWizard's internal database. The database is scanned to find one or more PBX definitions that match all recorded tones (i.e., dial tone, busy tone, ringing tone, reorder tone and any other manually-recorded tone - all match the definitions of the PBX). If a match is found, the entire PBX definition is reported (as comments) in the *ini* file using the same format.

Below shows an example of a file with full PBX/Country Database match:

```
## Some tones matched PBX/country Audc US
## Additional database tones guessed below (remove #'s to use).
#
# # Audc US, US Ringback tone
# [CALL PROGRESS TONE #5]
# Tone Type=2
# Low Freq [Hz]=450
# High Freq [Hz]=500
# Low Freq Level [-dBm]=0
# High Freq Level [-dBm]=0
# First Signal On Time [10msec]=180
# First Signal Off Time [10msec]=450
# Second Signal On Time [10msec]=0
# Second Signal Off Time [10msec]=0
```

Notes:

- If a match is found in the database, consider using the database's definitions instead of the recorded definitions, as they might be more accurate.
- For full operability of the FXO device, it may be necessary to edit this file and add more Call Progress Tone definitions. Sample Call Progress Tones *ini* files are available in the release package.
- When the call progress tones *ini* is complete, the corresponding dat file is ready for download. After loading this file to the device, repeat the automatic detection phase discussed above, and verify that the device detects all four call progress tones correctly.
- Manually changing the *ini* file causes the dat file to be outdated and it therefore, needs to be re-generated according to the new *ini* file. A dat file may be regenerated by clicking the **Regenerate** button at the final dialog or by using the DConvert utility.

10.3.6 Adding a Reorder Tone to the CPT File

The following procedure describes how to add a Reorder tone that a PBX generates to indicate a disconnected call, to the CPT file.

➢ **To add a Reorder tone to the CPT file:**

1. Make a call (using G.711) between the device FXO, which is connected to the PBX, and a remote entity in the IP network.
2. Capture the call using a network sniffer such as Whiteshark.
3. Disconnect the call from the PBX side, and then wait approximately 30 seconds before stopping the Whiteshark recording.
4. In the network trace, locate the RTP stream sent from the FXO.
5. Save the RTP payload on your PC as a *.pcm file by clicking **Save Payload** (**Statistics** menu > **RTP** > **Stream Analysis**). (Note: ensure that you select the 'forward' option.)
6. Open the *.pcm file in a voice recording utility such as CoolEdit.
7. Locate the tone that the PBX played to indicate the disconnected call (if such a tone exists).
8. Locate the attributes of the tone -- its frequency and interval (on / off time).
9. In the Call Progress Tones file, add a new Reorder Tone with the attributes you found in the previous step. Ensure that you update the numbers of the successive tones and the total number of tones in the beginning of the file.
10. Create a Call Progress Tones.dat file using the DConvert Utility (see "TrunkPack Downloadable Conversion Utility" on page 195).
11. Load the new file to the device, and then reset the device.

Reader's Notes

11 Diagnostics

Several diagnostic tools are provided to enable you to identify correct functioning of the device or an error condition with a probable cause and a solution or workaround.

The diagnostic tools include the following:

- Chassis LEDs - refer to the *Hardware Installation Manual*
- Self-Testing on hardware initialization – see "Self-Testing" on page 219
- FXS Line testing – see "FXS Line Testing" on page 220
- Error / notification messages via the following interfaces:
    - **Syslog:** Log messages can be viewed using an external Syslog server (see "Syslog Support" on page 221) or in the 'Message Log' page of the Web interface (see Activating the Internal Syslog Viewer). Note that the 'Message Log' page is not recommended for prolong debugging.
    - RS-232 terminal (for establishing a serial communications link with the device. (Applicable only to Analog devices.)
- Debug Recording using CLI – see "Debug Recording (DR)" on page 231

11.1 Self-Testing

The device features the following self-testing modes to identify faulty hardware components:

- **Startup Tests:** These tests have minor impact in real-time. While the Startup tests are executed, the regular operation of the device is disabled. If an error is detected, an error message is sent to the Syslog. The following hardware components are tested:
    - **CPU speed** - applicable only to 3000 Series
    - **TSA** (Time Slot Assigner) - applicable only to Digital PSTN
    - **CPU Version** - applicable only to 2000 Series
    - **PSTN framers** (when used) - applicable only to Digital PSTN
    - **Missing DSP's** - applicable only to Digital PSTN
    - **Lattice TPM and TER** - applicable only to 3000 Series devices
    - **GB Ethernet** - applicable only to 3000/6310 Series devices
    - **Voice path** - applicable only to Digital PSTN
- **Periodic Test:** (Applicable only to 2000 and 3000 Series devices.) Monitors the device during run-time. This test is performed after startup, even when there is full traffic on the device (quality is not degraded). This is a short test phase in which the only error detected and reported is failure in initializing hardware components or malfunction in running hardware components. If an error is detected, an error message is sent to the Syslog. The following hardware components are tested:
    - **Time Slot Assigner** (TSA) - applicable only to Digital PSTN
    - **PSTN framers** (when used) - applicable only to Digital PSTN
    - **Missing DSP's** - applicable only to Digital devices
    - **Lattice TPM and TER** - applicable only to 3000 Series devices

- **Gb Ethernet ports** - applicable only to 3000/6310 Series devices
- **Voice path** - applicable only to 3000 Series and 2000 Series devices for redundant blade

- **User-Initiated (Detailed) Test:** initiated by the user when the device is offline (not used for regular service). This test is used in addition to the Startup tests. The test is performed on startup when initialization of the device completes and if the parameter EnableDiagnostics is set to 1 or 2 (refer to the device's *User's Manual* for a description of his parameter). For Analog devices, the **Ready** and **Fail** LEDs are lit while the Detailed test is running. The following hardware components are tested:
  - **RAM** (when EnableDiagnostics = 1 or 2) - applicable only to Digital devices
  - **Flash memory** (when EnableDiagnostics = 1 or 2)
  - **DSPs** (when EnableDiagnostics = 1 or 2)
  - **Physical Ethernet ports** (when EnableDiagnostics = 1 or 2)
  - **Analog interfaces** (when EnableDiagnostics = 1 or 2) - applicable only to Analog devices
  - **UTOPIA Bridge** (when EnableDiagnostics = 2) - applicable only to 3000 Series devices

Notes:

- To continue regular operation, disable the Detailed test. Set the parameter EnableDiagnostics to 0, and then reset the device.
- When the Detailed test is enabled, ignore errors sent to the Syslog server.

11.2 Analog Line Testing

Note: This section is applicable only to Analog devices.

Analog Line testing is executed using SNMP only:

- For FXO interfaces: acAnalogFxoLineTestTable SNMP table
- For FXS interfaces: acAnalogFxsLineTestTable SNMP table

The device features a mechanism that performs tests on the telephone lines connected to FXS and FXO ports. These tests provide various line measurements. In addition to these tests (detailed below), a keep-alive test is also preformed every 100 msec on each of the analog ports to detect communication problems with the analog device and overheating (in FXS ports).

The following line tests are available on FXS interfaces:

- Hardware revision number
- Temperature (above or below limit, only if a thermometer is installed)
- Hook state
- Coefficients checksum

- Message waiting indication status
- Ring state
- Reversal polarity state
- Line current (only on port 0)
- Line voltage between TIP and RING (only on port 0)
- 3.3 V reading (only on port 0)
- Ring voltage (only on port 0)
- Long line current (only on port 0)

The following line tests are available on FXO interfaces:

- Line Current (mA)
- Line Voltage (V)
- Hook (0 = on-hook; 1 = off-hook)
- Ring (0 - Off; 1 - On)
- Line Connected (0 = Disconnected; 1 = Connected)
- Polarity state (0 = Normal; 1 = Reversed, 2 = N\A)
- Line polarity (0 = Positive; 1 = Negative)
- Message Waiting Indication (0 = Off; 1 = On)

Note: he line testing mechanism must be used only for monitoring and never when there are calls in progress.

11.3 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 Syslog protocol is defined in the IETF RFC 3164 standard.

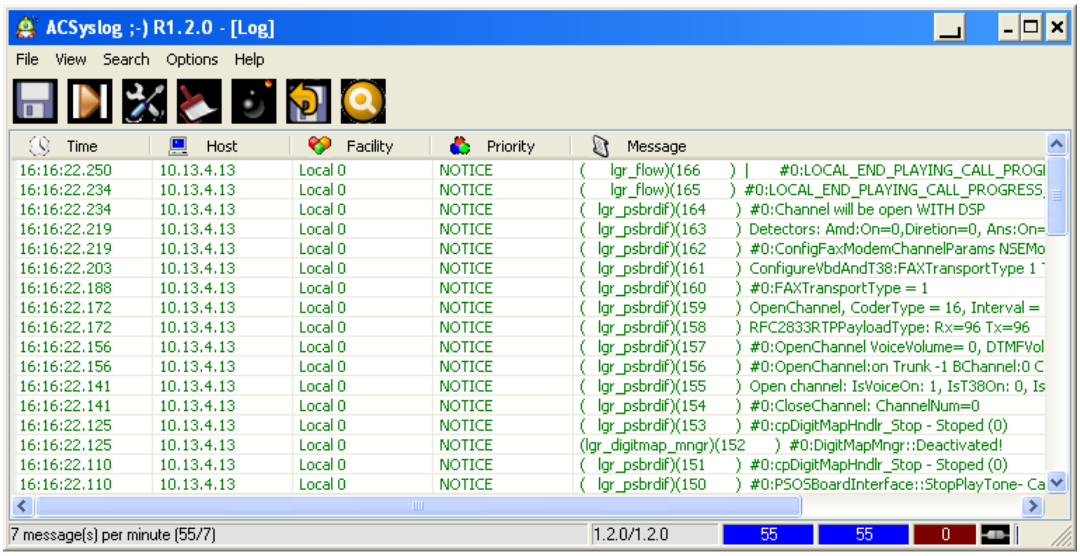
The device contains an embedded Syslog client,, which sends error reports / events that it generates to a remote Syslog server using the IP / UDP protocol. This information is a collection of error, warning, and system messages that records every internal operation of the device.

11.3.1 Syslog Servers

You can use any of the following Syslog server types for receiving Syslog messages generated by the device:

- **ACSyslog Program:** AudioCodes proprietary Syslog server, supplied with your device. The figure below displays an example of received Syslog messages in the ACSyslog program.

Figure 11-1: AudioCodes Proprietary Syslog Server - ACSyslog



Time	Host	Facility	Priority	Message
16:16:22.250	10.13.4.13	Local 0	NOTICE	(lgr_flow)(166) | #0:LOCAL_END_PLAYING_CALL_PROGI
16:16:22.234	10.13.4.13	Local 0	NOTICE	(lgr_flow)(165) #0:LOCAL_END_PLAYING_CALL_PROGRESS
16:16:22.234	10.13.4.13	Local 0	NOTICE	(lgr_psbrdif)(164) #0:Channel will be open WITH DSP
16:16:22.219	10.13.4.13	Local 0	NOTICE	(lgr_psbrdif)(163) Detectors: Amd:On=0,Diretion=0, Ans:On=
16:16:22.219	10.13.4.13	Local 0	NOTICE	(lgr_psbrdif)(162) #0:ConfigFaxModemChannelParams NSEMo
16:16:22.203	10.13.4.13	Local 0	NOTICE	(lgr_psbrdif)(161) ConfigureVbdAndT38:FAXTransportType 1
16:16:22.188	10.13.4.13	Local 0	NOTICE	(lgr_psbrdif)(160) #0:FAXTransportType = 1
16:16:22.172	10.13.4.13	Local 0	NOTICE	(lgr_psbrdif)(159) OpenChannel, CoderType = 16, Interval =
16:16:22.172	10.13.4.13	Local 0	NOTICE	(lgr_psbrdif)(158) RFC2833RTPPayloadType: Rx=96 Tx=96
16:16:22.156	10.13.4.13	Local 0	NOTICE	(lgr_psbrdif)(157) #0:OpenChannel VoiceVolume= 0, DTMFVol
16:16:22.156	10.13.4.13	Local 0	NOTICE	(lgr_psbrdif)(156) #0:OpenChannel:on Trunk -1 BChannel:0 C
16:16:22.141	10.13.4.13	Local 0	NOTICE	(lgr_psbrdif)(155) Open channel: IsVoiceOn: 1, IsT38On: 0, Is
16:16:22.141	10.13.4.13	Local 0	NOTICE	(lgr_psbrdif)(154) #0:CloseChannel: ChannelNum=0
16:16:22.125	10.13.4.13	Local 0	NOTICE	(lgr_psbrdif)(153) #0:cpDigitMapHndlr_Stop - Stoped (0)
16:16:22.125	10.13.4.13	Local 0	NOTICE	(lgr_digitmap_mngr)(152) #0:DigitMapMngr::Deactivated!
16:16:22.110	10.13.4.13	Local 0	NOTICE	(lgr_psbrdif)(151) #0:cpDigitMapHndlr_Stop - Stoped (0)
16:16:22.110	10.13.4.13	Local 0	NOTICE	(lgr_psbrdif)(150) #0:PSOSBoardInterface::StopPlayTone- Ca

7 message(s) per minute (55/7) 1.2.0/1.2.0 55 55 0

- **Embedded Syslog Server:** The device provides an embedded Syslog server, which is accessed through the Web interface. This provides limited Syslog server functionality. For more information, see Section 11.3.2.5.

- **Wireshark:** Third-party network protocol analyzer (http://www.wireshark.org).

- Any third-party, Syslog server. A typical Syslog server program enables filtering of messages according to parameters such as priority, IP sender address, time, and date.

11.3.2 Syslog Message Format

The Syslog message is transmitted 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 (using the SyslogServerPort parameter).

Syslog generates the following types of messages:

- **Error:** indicates a problem has been identified that requires immediate handling
- **Warning:** indicates an error might occur if measures are not taken to prevent it
- **Notice:** indicates an unusual event has occurred
- **Info:** indicates an operational message
- **Debug:** messages used for debugging

When using the device's embedded Syslog server, these message types are color coded (as explained in Section 11.3.2.5).

Note: The Info and Debug Syslog messages are required only for advanced debugging. Therefore, by default, tthey are not sent by the device.

Syslog messages received from the SIP application level are sequentially numbered. A skip in the sequence of messages indicates a loss of SIP message packets. For example, in the below Syslog message generation, SIP messages 622 through 629 were not received. In other words, 9 Syslog messages were lost (the sequential numbers are indicated below in **bold** font):

```
18:38:14. 52 : 10.33.45.72 : NOTICE: (lgr_psbrdex)(619) recv <--
DIGIT(0) Ch:0 OnTime:0 InterTime:100 Direction:0 System:1 [File:
Line:-1]
18:38:14. 83 : 10.33.45.72 : NOTICE: (lgr_flow)(620)
#0:DIGIT_EV [File: Line:-1]
18:38:14. 83 : 10.33.45.72 : NOTICE: (lgr_flow)(621)  |
#0:DIGIT_EV [File: Line:-1]
18:38:14.958 : 10.33.45.72 : NOTICE: (lgr_flow)(630)  |
#0:DIGIT_EV [File: Line:-1]
```

Note: When Network Time Protocol (NTP) is enabled, a timestamp string **[hour:minutes:seconds]** is added to all Syslog messages (for information on NTP, refer to the device's *User's Manual*).

11.3.2.1 Event Representation in Syslog Messages

The Syslog message events that the device sends are represented by unique abbreviations. An example of an abbreviated event in a Syslog message indicating packet loss (PL) is shown below:

```
Apr  4 12:00:12 172.30.1.14 PL:5  [Code:3a002] [CID:3294] [Time:
20:17:00]
```

The table below lists the unique event abbreviations:

Table 11-1: Syslog Error Name Descriptions

Error Abbreviation	Error Name Description
AA	Invalid Accumulated Packets Counter
AC	Invalid Channel ID
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
OR	DSP JB Overrun
PH	Packet Header Error
PL	RTP Packet Loss
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
AA	Invalid Accumulated Packets Counter
AC	Invalid Channel ID
AL	Invalid Header Length

Error Abbreviation	Error Name Description
AO	Invalid Codec Type
AP	Unknown Aggregation Payload Type
AR	Invalid Routing Flag Received

11.3.2.2 Unique Device Identification in Syslog Messages

For MSBG and Mediant 3000 devices, the Syslog messages include a unique string to identify these devices:

- **Mediant 800 MSBG and Mediant 1000 MSBG:** Syslog messages relating to VoIP functionality are marked with "host"; those relating to Data Routing are marked with "**DATA**".

```
12/12 12:46:40.921 : 10.8.5.70 : NOTICE  : host:  10.8.5.78
(sip_stack)(24)  Resource SIPMessage deleted - #267
11/24 08:14:09.311 : 10.3.2.100 : WARNING : DATA: Failed to
set device eth0 netmask: Cannot assign requested address
```

- **Mediant 3000:** High Availability (HA) main operations and events are sent to the Syslog with the prefix "**M3K_HA**". All Syslog messages and events of the redundant TP-6310 blade are sent to the Syslog by the active TP-6310 blade with the "**Redundant module message**" message prefix.

11.3.2.3 Identifying AudioCodes Syslog Messages using Facility Levels

The device’s Syslog messages can easily be identified and distinguished from other Syslog messages by setting its Facility level. The Facility levels of the device's Syslog messages are numerically coded with decimal values. Facility level may use any of the "local use" facilities (0 through 7), according to RFC 3164.

Implementing Facility levels is useful, for example, if you collect the device’s as well as other equipments’ Syslog messages, on a single server. Therefore, in addition to filtering Syslog messages according to IP address, the messages can be filtered according to Facility level.

The Facility level is configured using the *SyslogFacility* parameter, which provides the following options:

Table 11-2: Syslog Facility Levels

Numerical Value	Facility Level
16 (default)	local use 0 (local0)
17	local use 1 (local1)
18	local use 2 (local2)
19	local use 3 (local3)
20	local use 4 (local4)
21	local use 5 (local5)

Numerical Value	Facility Level
22	local use 6 (local6)
23	local use 7 (local7)

Syslog messages begin with a less-than ("**<**") character, followed by a number, which is followed by a greater-than ("**>**") character. This is optionally followed by a single ASCII space. The number is known as the *Priority* and represents both the Facility level and the Severity level. A Syslog message with Facility level 16 is shown below:

```
Facility: LOCAL0 - reserved for local use (16)
```

11.3.2.4 SNMP Alarms in Syslog Messages

SNMP is a protocol that alerts you when a network-attached device requires attention. 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 11-3: 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:

11.3.2.5 Viewing Syslog Messages in the Web Interface

The Web interface's 'Message Log' page displays Syslog messages sent by the device.

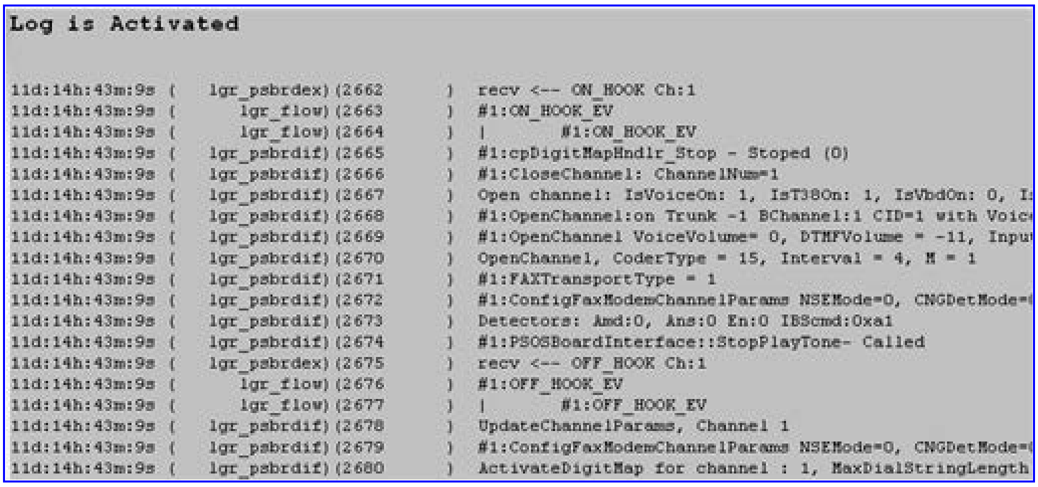
Notes:

- 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.
- Syslog message display through the Web interface is currently not supported on the Mediant 800 MSBG and Mediant 1000 MSBG devices.

➢ To view Syslog messages in the Web interface:

1. Acces the Web interface.
2. Enable the device's Syslog feature and configure the Syslog parameters (see Section 11.3.3).
3. Open the Message Log page (**Status & Diagnostics** tab > **System Status** menu > **Message Log**); the Message Log page is displayed and the Syslog is activated.

Figure 11-2: Viewing Syslog Messages in the Web Interface



```
Log is Activated


11d:14h:43m:9s (   lgr_psbrdex)(2662      )  recv <-- ON_HOOK Ch:1
11d:14h:43m:9s (      lgr_flow)(2663      )  #1:ON_HOOK_EV
11d:14h:43m:9s (      lgr_flow)(2664      )  |       #1:ON_HOOK_EV
11d:14h:43m:9s (   lgr_psbrdif)(2665      )  #1:cpDigitMapHndlr_Stop - Stoped (0)
11d:14h:43m:9s (   lgr_psbrdif)(2666      )  #1:CloseChannel: ChannelNum=1
11d:14h:43m:9s (   lgr_psbrdif)(2667      )  Open channel: IsVoiceOn: 1, IsT38On: 1, IsVbdOn: 0, I
11d:14h:43m:9s (   lgr_psbrdif)(2668      )  #1:OpenChannel:on Trunk -1 BChannel:1 CID=1 with Voic
11d:14h:43m:9s (   lgr_psbrdif)(2669      )  #1:OpenChannel VoiceVolume= 0, DTMFVolume = -11, Inpu
11d:14h:43m:9s (   lgr_psbrdif)(2670      )  OpenChannel, CoderType = 15, Interval = 4, M = 1
11d:14h:43m:9s (   lgr_psbrdif)(2671      )  #1:FAXTransportType = 1
11d:14h:43m:9s (   lgr_psbrdif)(2672      )  #1:ConfigFaxModemChannelParams NSEMode=0, CNGDetMode=
11d:14h:43m:9s (   lgr_psbrdif)(2673      )  Detectors: Amd:0, Ans:0 En:0 IBScmd:0xa1
11d:14h:43m:9s (   lgr_psbrdif)(2674      )  #1:PSOSBoardInterface::StopPlayTone- Called
11d:14h:43m:9s (   lgr_psbrdex)(2675      )  recv <-- OFF_HOOK Ch:1
11d:14h:43m:9s (      lgr_flow)(2676      )  #1:OFF_HOOK_EV
11d:14h:43m:9s (      lgr_flow)(2677      )  |       #1:OFF_HOOK_EV
11d:14h:43m:9s (   lgr_psbrdif)(2678      )  UpdateChannelParams, Channel 1
11d:14h:43m:9s (   lgr_psbrdif)(2679      )  #1:ConfigFaxModemChannelParams NSEMode=0, CNGDetMode=
11d:14h:43m:9s (   lgr_psbrdif)(2680      )  ActivateDigitMap for channel : 1, MaxDialStringLength
```

The displayed logged messages are color coded as follows:

- **Yellow:** Error messages
- **Blue:** Recoverable error messages
- **Black:** Notice messages

4. To stop or clear the Syslog messages, open a different Web page. If you want to start a new Syslog session, re-access the 'Message Log' page (see Step 3).

11.3.3 Configuring the Syslog Feature

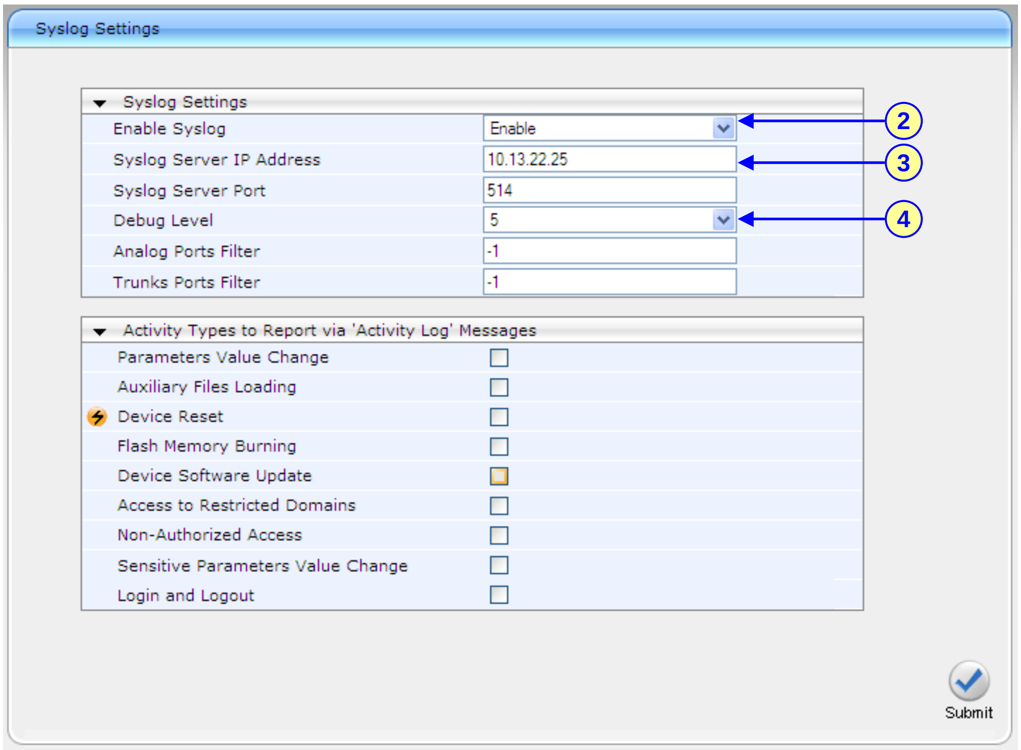
The Syslog client, which is embedded in the device, sends error reports/events generated by the device to a Syslog server using IP/UDP protocol. The Syslog can be configured using the Web interface, EMS, or *ini* file. The main configuration of the Syslog feature includes enabling the Syslog client, defining the Syslog server's IP address, and then selecting the debug level.

The procedure below describes how to configure Syslog for receiving SIP messages through the Web interface.

➢ To configure Syslog for receiving SIP message events:

1. Acces the Web interface.
2. Open the Syslog Settings page (**Configuration** tab > **System** menu > **Syslog Settings**).

Figure 11-3: Configuring Syslog (for SIP Messages) in the Web Interface



Syslog Settings

▼ Syslog Settings

Enable Syslog	Enable
Syslog Server IP Address	10.13.22.25
Syslog Server Port	514
Debug Level	5
Analog Ports Filter	-1
Trunks Ports Filter	-1

▼ Activity Types to Report via 'Activity Log' Messages

Parameters Value Change	□
Auxiliary Files Loading	□
Device Reset	□
Flash Memory Burning	□
Device Software Update	□
Access to Restricted Domains	□
Non-Authorized Access	□
Sensitive Parameters Value Change	□
Login and Logout	□

3. From the 'Enable Syslog' drop-down list, select 'Enable'.
4. In the 'Syslog Server IP Address' field, enter the IP address of the Syslog server (or the IP address of the computer on which the Syslog server is running).
5. From the 'Debug Level' drop-down list, select '5'.
6. Click the **Submit** button to apply your settings.

Notes: In addition to the settings described in the procedure above, the device provides additional, optional Syslog configuration parameters (for more information, refer to the *User's Manual*).

12 Debug Recording

The Debug Recording (DR) mechanism duplicates all messages that are sent and/or received by the device and sends them to an external IP address. It is used for advanced debugging when it is required to analyze internal messages and signals. In addition, DR is useful for recording network traffic in environments in which hub / port mirroring isn’t available and to record internal traffic between two endpoints on the same gateway.

DR can be used to capture the following message types:

- Digital signal processor (DSP) recording (see Section 12.2.1 on page 231):
  - RTP/RTCP streams that are sent and/or received by the device.
  - Voice signals (pulse-code modulation / PCM):
    - From the PSTN/PBX, before it enters the DSP.
    - Sent from the DSP to the PSTN/PBX.
  - Other internal information (such as DSP events and commands).
- PSTN trace - received/transmitted ISDN and CAS messages (see Section 12.2.2 on page 233).
- Control messages - SIP, MGCP, MEGACO, and TPNCP (see Section 12.2.3 on page 236).
- Networking streams (such as T.38, HTTP and SCTP (see Section 12.2.4 on page 237).

Notes:

- DSP, PSTN, Control and IP recording can be performed simultaneously.
- All DR rules that are defined through the CLI are deleted after the device is reset.
- DR can be used on a “live” device for debugging. For normal operation (unless otherwise requested by AudioCodes’ support), DR must be disabled.
- DR doesn’t require DSP channels and therefore, can be used when the device operates at full capacity.

12.1 Collecting DR Messages

The client that is used to capture the DR packets is the open source Wireshark program. This program can be downloaded from www.wireshark.org. In addition, AudioCodes proprietary plugin files (supplied in the software kit) must be located in the *plugin* folder of the installed Wireshark version program (typically, *C:\Program Files\WireShark\plugins\<Wireshark version>*).

The default DR port is 925. This can be changed in Wireshark (**Edit** menu > **Preferences** > **Protocols** > **ACDR**). When loaded, the Wireshark plugin dissects all packets on port 925 as DR packets.

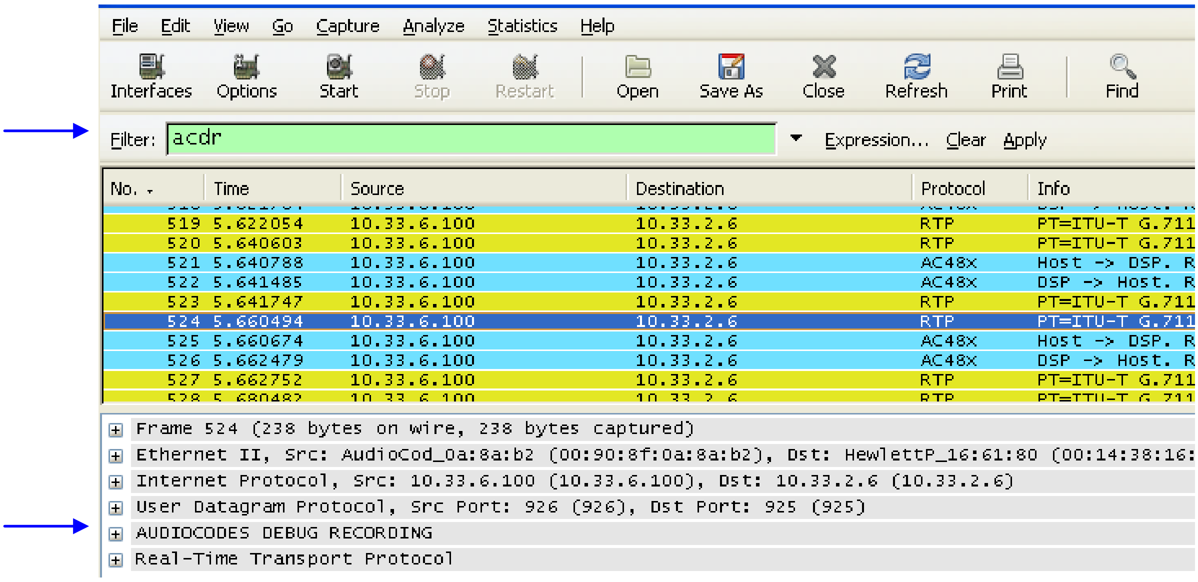
Notes:

- The plugins for DR are per major software release.
- The plugins are applicable to Wireshark Version 99.08.
- The plugins are backward compatible.
- From Wireshark version 99.08, the **tpncp.dat** file must be located in the folder *C:\Program Files\WireShark\tpncp*.

12.1.1 Viewing DR Messages in Wireshark

Use the **acdr** filter to view the DR messages in Wireshark. The source IP address of the DR messages is always the OAMP IP address of the device. The DR mechanism adds to each message the proprietary header, “AUDIOCODES DEBUG RECORDING”.

Figure 12-1: Viewing DR Messages in Wireshark



Filter: acdr

No.	Time	Source	Destination	Protocol	Info
519	5.622054	10.33.6.100	10.33.2.6	RTP	PT=ITU-T G.711
520	5.640603	10.33.6.100	10.33.2.6	RTP	PT=ITU-T G.711
521	5.640788	10.33.6.100	10.33.2.6	AC48x	Host -> DSP. R
522	5.641485	10.33.6.100	10.33.2.6	AC48x	DSP -> Host. R
523	5.641747	10.33.6.100	10.33.2.6	RTP	PT=ITU-T G.711
524	5.660494	10.33.6.100	10.33.2.6	RTP	PT=ITU-T G.711
525	5.660674	10.33.6.100	10.33.2.6	AC48x	Host -> DSP. R
526	5.662479	10.33.6.100	10.33.2.6	AC48x	DSP -> Host. R
527	5.662752	10.33.6.100	10.33.2.6	RTP	PT=ITU-T G.711
528	5.680482	10.33.6.100	10.33.2.6	RTP	PT=ITU-T G.711

```
⊞ Frame 524 (238 bytes on wire, 238 bytes captured)
⊞ Ethernet II, Src: AudioCod_0a:8a:b2 (00:90:8f:0a:8a:b2), Dst: HewlettP_16:61:80 (00:14:38:16:
⊞ Internet Protocol, Src: 10.33.6.100 (10.33.6.100), Dst: 10.33.2.6 (10.33.2.6)
⊞ User Datagram Protocol, Src Port: 926 (926), Dst Port: 925 (925)
⊞ AUDIOCODES DEBUG RECORDING
⊞ Real-Time Transport Protocol
```

12.2 Debug Recording Modes

This section describes the different DR modes of operation.

12.2.1 DSP Recording

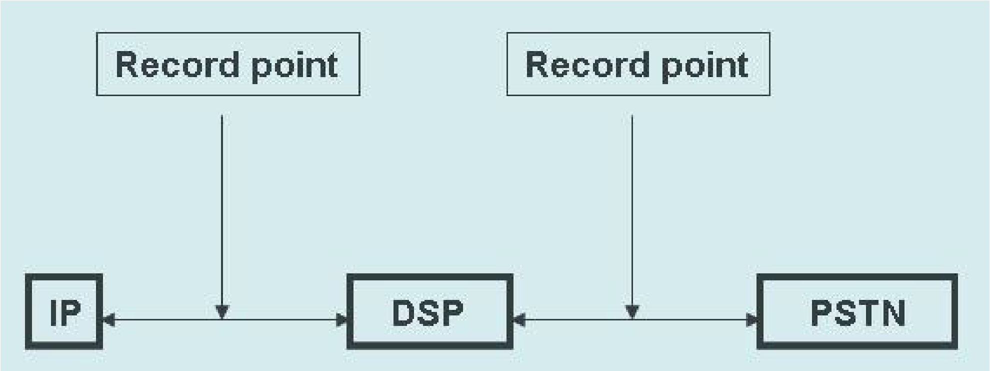
DSP recording should be used for analyzing voice-related issues such as: poor voice quality, echo, and fax / modem transmission.

The following messages are recorded **per channel**:

- Internal DSP packets and events.
- Incoming and outgoing RTP / RTCP streams (in the actual voice coder that is used by the channel).
- PCM - voice signal that arrives from and sent to the TDM (before it enters the DSP). The two streams are sent in G.711 A-law.

Notes: DSP recording cannot be used to record T.38 messages. To record T.38 messages, use the **AddIPTrafficTrace** command (see Section 12.2.4 on page 237)

Figure 12-2: DSP Record Points



When DSP recording is performed, for each call there are four media streams:

- **Network to DSP:** RTP messages received by the device
- **DSP to Network:** RTP messages sent by the device
- **TDM to DSP:** Voice signals received by the device from the PSTN/PBX
- **DSP to TDM:** Voice signals sent to the PBX/PSTN

Below is an example of the AUDIOCODES DEBUG RECORDING header that is used in DSP recording:

```
         AUDIOCODES DEBUG RECORDING
             Version: 0x01
             Time Stamp: 0000EA32C298(3929.195160 sec)
             Source ID: 0
             Dest ID: 0
             Reserved: AA
             Trace Point: Dsp -> Network (1)
             Media Type: RTP Packet (1)
             Payload offset: 9
             Header Extension
                 Packet destination IP address: 10.33.6.100
         (10.33.6.100)
                 Packet destination UDP port: 6010
                 Packet source UDP port: 6000
                 IP type of service: 184
```

12.2.1.1 DSP Recording for Mediant 800 MSBG

DSP recording for Mediant 800 includes different trace points. The voice path is:

Network <-> **VOIP encoder\decoder** <-> **DSP encoder\decoder** <-> **TDM**

Therefore, DSP recording for Mediant 800 provides four additional trace points. However, as they provide some common information, the recording produces two main traces:

- From the network, there are trace points “before VOIP decoder” (trace point #20) and “before DSP decoder” (18) that currently represent the same stream.
- From the Tel side, there are trace points “before VOIP encoder” (21) and “before NET encoder” (22).

These four trace points replace the DSP to network, and network to DSP.

12.2.1.2 Activating DSP Recording

You can activate DSP recording by using one of the following command options:

- **AddNextCallTrace:** Records the next *x* number of media calls.
- **AddTrunkBchannelTrace:** Records media calls according to trunk and B-channel (applicable only to digital PSTN interfaces).
- **AddChannelIdTrace:** Records media calls according to Channel ID (CID).

The **AddNextCallTrace** command is the most useful one to perform DSP recording when there are limited number of calls on the device. It cannot be used, for example, when a specific call needs to be recorded on a device that handles many dozens of calls. In this case, you need to isolate the problematic call on a specific Trunk/B-channel and use the **AddTrunkBchannelTrace** or **AddChannelIdTrace** commands.

12.2.2 PSTN Traces

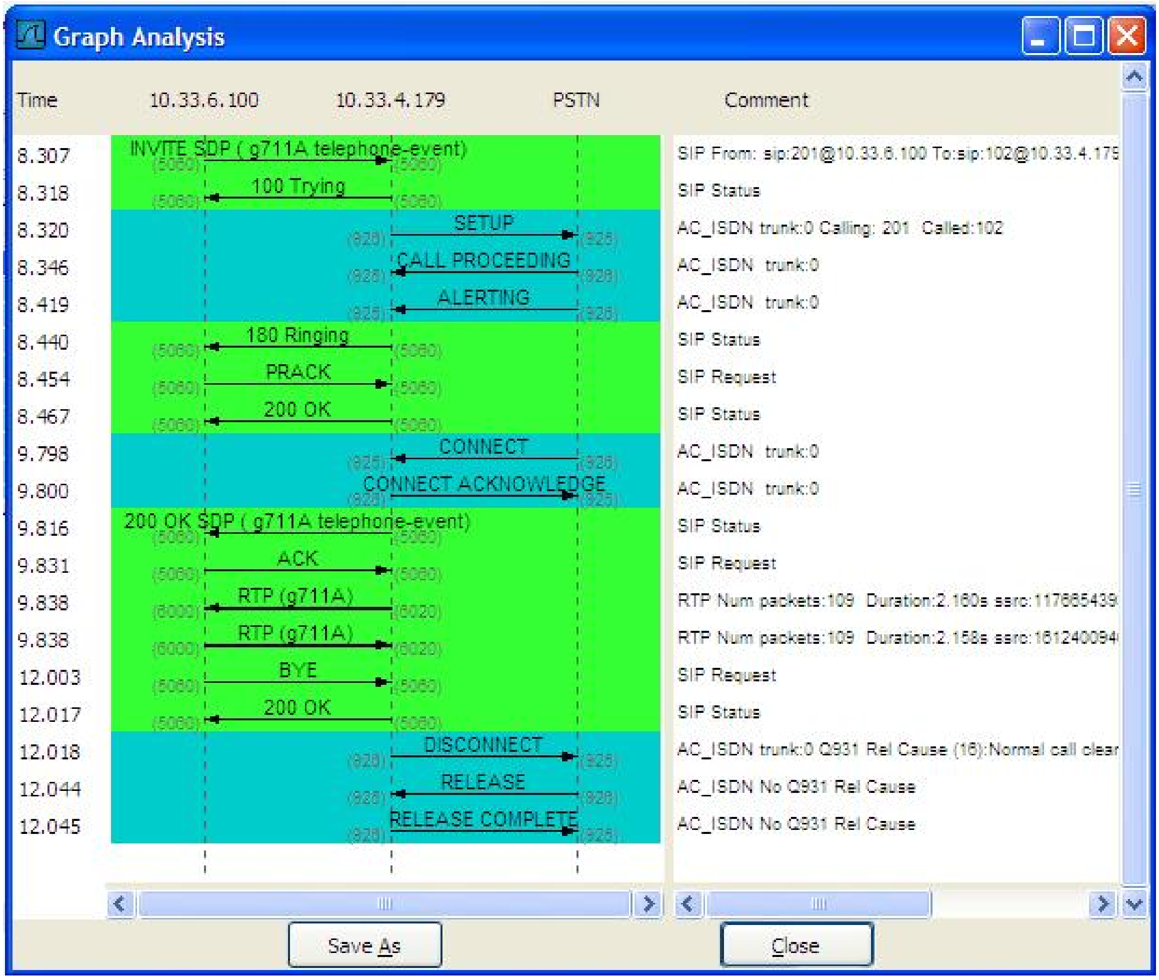
PSTN traces record ISDN and CAS messages that are exchanged between the device and the PSTN/PBX switch.

Notes: PSTN trace messages can be sent through debug recording or directly to the Syslog. To send PSTN messages to the Syslog server, add the following parameter setting **PSTNReserved3=8** to the *ini* file, and then reset the device.

12.2.2.1 ISDN Traces

ISDN call flow can be viewed using the Wireshark's **q931** filter. Wireshark also allows you to convert the trace to a graph (**Statistics** > **VoIP Calls**) and viewed along with SIP messages.

Figure 12-3: ISDN Trace Graph in Wireshark



Time		Comment
8.307	INVITE SDP (g711A telephone-event)	SIP From: sip:201@10.33.6.100 To:sip:102@10.33.4.179
8.318	100 Trying	SIP Status
8.320	SETUP	AC_ISDN trunk:0 Calling: 201 Called:102
8.346	CALL PROCEEDING	AC_ISDN trunk:0
8.419	ALERTING	AC_ISDN trunk:0
8.440	180 Ringing	SIP Status
8.454	PRACK	SIP Request
8.467	200 OK	SIP Status
9.798	CONNECT	AC_ISDN trunk:0
9.800	CONNECT ACKNOWLEDGE	AC_ISDN trunk:0
9.816	200 OK SDP (g711A telephone-event)	SIP Status
9.831	ACK	SIP Request
9.838	RTP (g711A)	RTP Num packets:109 Duration:2.160s ssrc:117665439
9.838	RTP (g711A)	RTP Num packets:109 Duration:2.158s ssrc:161240094
12.003	BYE	SIP Request
12.017	200 OK	SIP Status
12.018	DISCONNECT	AC_ISDN trunk:0 Q931 Rel Cause (16):Normal call clear
12.044	RELEASE	AC_ISDN No Q931 Rel Cause
12.045	RELEASE COMPLETE	AC_ISDN No Q931 Rel Cause

For ISDN messages, the additional header, “NetBricks Trace” is added below “AUDIOCODES DEBUG RECORDING”. The example below shows an example of an ISDN trace with this header:

```
AUDIOCODES DEBUG RECORDING
NetBricks Trace
System time: 3559
    Direction: Message received from internal server queue
(73)
From (Entity origination ID): DL_D (DL LAPD Q.921) (100)
    To (Entity destination ID): PH_D (D channel physical)
(68)
    Primitive code: 67
    NAI (Network Access ID): 0 -> number of trunk
    SAPI: 1
    Connection ID: 0
    Congestion flag: 0
    Allocated message: 2
    Allocated buffer: 3
    Allocated timer cell: 141
    IT Message stack counter: 120
    IT Buffer stack counter: 120
    Message congestion counter: 0
    Buffer congestion counter: 0
    IT Stack message congestion counter: 0
    IT Stack buffer congestion counter: 0
    Pointer to message: 689
    Pointer to buffer: 0
    Data size: 33
    Link Access Procedure, Channel D (LAPD)
    Q.931
        Protocol discriminator: Q.931
        Call reference value length: 2
        Call reference flag: Message sent from
originating side
        Call reference value: 0300 - > can be used as a
filter to  identify entire ISDN call
        Message type: SETUP (0x05)
        Bearer capability
        Channel identification
        Calling party number: '201'
        Called party number: '102'
        Sending complete
```

12.2.2.2 CAS Traces

CAS traces include the ”CAS Trace” header. The example below shows an example of a CAS trace, showing this header:

```
AUDIOCODES DEBUG RECORDING
CAS Trace
    Timer: 1145504439
    From: DSP (0)
    Current State: 7
    Event: EV_DIAL_ENDED (15)
    Next State: -1
    Function Use: Unknown (-1)
         Parameter 1: -1
    Parameter 2: -1
    Parameter 3: -1
    Trunk Number: 3
    BChannel Number: 23
    Call Handle: 0
```

12.2.2.3 Enabling PSTN Traces per Trunk

By default, PSTN traces are disabled. You can enable PSTN traces for a specific trunk, using the device's Web interface, CLI, or *ini* file.

12.2.2.3.1Using the Web Interface

The procedure below describes how to enable PSTN traces per trunk, using the Web interface.

➢ **To enable PSTN traces for a specific trunk, using the Web interface:**

1. Access the 'Trunk Settings' page (**Configuration** tab > **PSTN Settings** menu > **Trunk Settings**).
2. Stop the trunk, by clicking the **Stop Trunk** ■ button.
3. Click the **Advanced Parameter List** link (located on the top-right corner of the page).
4. From the 'Trace Level' drop-down list, select "Full ISDN Trace".
5. Click the **Apply Trunk Settings** ✓ button.

12.2.2.3.2Using the CLI

The procedure below describes how to enable PSTN traces per trunk, using the CLI.

➢ To enable PSTN traces for a specific trunk, using the CLI:

1. Access the CLI.
2. Enter the following commands:

```
pstn
PstnCOmmon
PstnSetTraceLevel <Trunk ID> -1 1
```

For example, to enable PSTN traces on the first Trunk, type the following:

```
PstnSetTraceLevel 0 -1 1
```

Note: For PSTN traces using CLI, you do not need to stop the trunk.

12.2.2.3.3Using the INI File

The procedure below describes how to enable PSTN traces per trunk, using the *ini* file.

➢ To enable PSTN traces for a specific trunk, using the ini file:

- Load an *ini* file to the device with the following *ini* file parameter settings:

```
TraceLevel_0 = 1
; where 0 depicts the first trunk number
```

12.2.3 Control Traces

Control traces are used to record incoming and/or outgoing control messages (i.e., SIP, MEGACO, MGCP or TPNCP).

12.2.4 Network Traces

Network traces are used to record any IP stream that isn't associated with media (RTP/RTCP) according to destination and/or source IP address or port and Layer-4 protocol (UDP, TCP, SCTP or any other IP type as defined by http://www.iana.com). Network traces are typically used to record T.38, SCTP or HTTP.

Note: To record T.38 messages, use the following commands:

- **AddIPTrafficTrace host2Net udp a a all all**
- **AddIPTrafficTrace Net2host udp a a all all**

12.3 Fast Track for DR

Debug Recording activation is performed using the CLI interface under the *DebugRecording* directory. This section describes the basic procedures for quickly activating DR and collecting call traces.

For a more detailed description of all the DR commands, see Section 0 on page 240.

12.3.1 Activating DR

The procedure below describes how to initially activate the DR. Once activated, you can perform the required traces or recordings, as described in the subsequent subsections.

➢ **To activate the DR:**

1. Start a CLI management session.
2. At the prompt, type the following command to access the DebugRecording directory:
   ```
   DR
   ```
3. At the prompt, type the following command to terminate all active recordings, if any:
   ```
   STOP
   ```
4. At the prompt, type the following command to remove all previous recording rules:
   ```
   RTR ALL
   ```
5. At the prompt, type the following command to remove all DR targets (i.e., client IP addresses) from the list:
   ```
   RT ALL
   ```
6. At the prompt, type the following command to define the IP address of the PC (running Wireshark) to where the device sends its debug packets:
   ```
   AIT <IP address of the target>
   ```
7. Continue with the procedures described below according to the required recording.

12.3.2 Generating DSP Traces

The procedure below describes how to generate DSP traces.

➢ To perform DSP traces:

1. Setup DR, as described in Section 12.3.1 on page 237.
2. At the prompt, type the following command so that the next call on the device is recorded:

```
ANCT ALL-WITH-PCM 1
```

3. At the prompt, type the following:

```
START
```

12.3.3 Generating ISDN Traces

The procedure below describes how to generate ISDN traces.

➢ To perform ISDN trace:

1. Setup DR, as described in Section 12.3.1 on page 237.
2. Enable PSTN traces for a specific trunk (see Section 12.2.2.3 on page 235).
3. At the prompt, type the following command:

```
APST ISDN
```

4. At the prompt, type the following command:

```
START
```

12.3.4 Capturing Control Packets

The procedure below describes how to capture control (i.e., SIP, MGCP, MEGACO, or TPNCP) packets.

➢ To capture Control packets

1. Setup DR, as described in Section 12.3.1 on page 237.
2. At the prompt, type the following command (or MGCP/MEGACO/TPNCP):

```
AddIPControlTrace n2h SIP
```

   Note that instead of SIP, you can type **MGCP**, **MEGACO**, or **TPNCP**.

3. At the prompt, type the following command:

```
START
```

12.3.5 Capturing T.38 Traffic

The procedure below describes how to capture T.38 traffic.

➢ **To capture T.38 traffic you need to record all UDP messages that are sent received by the device:**

1. Setup DR, as described in Section 12.3.1 on page 237.
2. At the prompt, type the following command:

```
AddIPTrafficTrace host2Net udp a a all all
```

3. At the prompt, type the following command:

```
AddIPTrafficTrace Net2host udp a a all all
```

4. At the prompt, type the following command:

```
START
```

12.3.6 Recording SCTP Traffic

The procedure below describes how to record Stream Control Transmission Protocol (SCTP) traffic.

➢ **To record SCTP traffic:**

1. Setup DR, as described in Section 12.3.1 on page 237.
2. At the prompt, type the following commands:

```
aiptt n2h 132 a a
aiptt h2n 132 a a
```

3. At the prompt, type the following command:

```
START
```

12.4 DR Command Reference

The tables below describe the DR commands. You can also view the description of a DR command in the CLI interface by simply typing the command name without any arguments.

12.4.1 Client Setup Commands

Table 12-1: Client Setup Commands

Command	Parameters	Description
AddIpTarget	IPAddr [UDPPort]	Adds a Wireshark DR IP client to the list. UDPPort (optional): port on which to send the recorded packets (default is 925).
RemoveTarget	Index	Removes a DR client from the list. Index: index for the removed target (as displayed via ListTargets).
ListTargets		Displays the client list.
SetDefaultTarget	Index	Changes the default target. The default target is the first target added (AddTarget). Index: index for the default target (as displayed via ListTargets).

12.4.2 Trace Rule Commands

Table 12-2: Trace Rule Commands

Command	Parameters	Description
AddIPTrafficTrace	TracePoint PDUType SourcePort DestPort [SourceIP] [DestIP] [DebugTarget]	Record IP traffic. ▪ **Trace Point:** ✓ Net2Host = Inbound non-media traffic. ✓ Host2Net = outbound non-media traffic. ▪ **PDUType:** ✓ UDP = UDP traffic. ✓ TCP = TCP traffic. ✓ ICMP = ICMP traffic. ✓ IPType = Any other IP type (as defined by http://www.iana.com). ✓ A = All traffic types. ▪ **SourcePort:** Datagram’s source port number (ALL for IP wildcard). ▪ **DestPort:** Datagram’s destination port number (ALL for IP wildcard). ▪ **SourceIP (optional):** Datagram’s source IP address (ALL for IP wildcard). ▪ **DestIP (optional):** Datagram’s source IP address (ALL for IP wildcard).

Command	Parameters	Description
		▪ **DebugTarget (optional):** Debug target list index; if not specified, the default target is used.
AddIPControlTrace	TracePoint ControlType [DebugTarget]	Records an IP control. ▪ **Trace Point:** ✓ Net2Host = Inbound/Outbound non-media traffic. ▪ **ControlType:** ✓ SIP = SIP traffic. ▪ **DebugTarget (optional):** Debug target list index; if not specified, the default target is used.
AddPstnSignaling Trace	PacketType [DebugTarget]	Records PSTN signaling. ▪ **Packet Type:** ✓ CAS = CAS signaling. ✓ ISDN = ISDN signaling. ▪ **DebugTarget (optional):** Debug target list index; if not specified, the default target is used. **Notes:** ▪ Applicable only to Digital PSTN devices. ▪ To record PSTN signaling, 'PSTN Trace Level' (TraceLevel ini file) must be set to 1.
AddNextCallTrace	PacketType NumOfCalls [TraceType] [DebugTarget]	Records the next media calls. ▪ **Packet Type:** ✓ ALL = All media-related (internal DSP packets, RTP, RTCP, T38, events) of a specific call. ✓ ALL-WITH-PCM = All media-related and PCM traffic of a specific call. ▪ **NumOfCalls:** Amount of next media calls to record. (**Note:** Currently, only one call can be recorded.) ▪ **Trace Type (optional):** ✓ New (default) = Next new NumOfCalls calls to record. When these calls end, new calls are not recorded. ✓ Dynamic = Next new NumOfCalls calls to record. When these calls end, new calls are recorded until this trace is deleted. ▪ **RemoteIPAddr:** Captures number (according to the 'NumOfCalls' parameter) of next call, but with the special condition that these next calls should use only the specified remote IP address. For example: "AddNextCallTrace All 10 Dynamic 10.31.2.85" In this example, the next 10 dynamic RTP calls that activate the RTP to a specific remote IP address (i.e., 10.31.2.85) are recorded.

Command	Parameters	Description
AddTrunkBchannelTrace	PacketType TRUNK [TO_TRUNK] [BCHANNEL] [TO_BCHANNEL][DebugTarget]	Records media calls according to trunk and B-channel. ▪ **Packet Type:** ✓ ALL = All media-related (internal DSP packets, RTP, RTCP, T38, events) of a specific call. ✓ ALL-WITH-PCM = All media-related and PCM traffic of a specific call. ▪ **Trunk:** Start of range trunk number for recording. (Note: Currently, only 1 channel can be recorded.) ▪ **To_Trunk (optional):** End of range trunk number. ▪ **BChannel (optional):** Start of range B-Channel number for recording. ▪ **To_BChannel (optional):** End of range B-Channel number for recording. ▪ **DebugTarget (optional):** Debug target list index; if not specified, the default target is used. **Note:** Applicable only to Digital PSTN devices.
AddChannelIdTrace	PacketType Channel-Id [To Channel-Id][DebugTarget]	Records media calls according to CID. ▪ **Packet Type:** ✓ ALL = All media-related (internal DSP packets, RTP, RTCP, T38, events) of a specific call. ✓ ALL-WITH-PCM = All media-related and PCM traffic of a specific call. ▪ **Channel-Id:** Start of range channel ID number for recording. (**Note:** Currently, only one channel can be recorded for digital devices.) ✓ To Channel-Id (optional) = End of range channel ID number for recording. ▪ **DebugTarget (optional):** Debug target list index; if not specified, the default target is used.
RemoveTraceRule	Index	Removes TraceRule from list. Index: Rule index (as displayed via ListTraceRules). ALL for rule wildcard.
ListTraceRules	-	Displays the following: ▪ Added TraceRules. ▪ Status of Debug Recording tool (Active or Inactive). ▪ Number of debug recording connections.

12.4.3 DR Activation Commands

Table 12-3: DR Activation Commands

Command	Parameters	Description
STARTrecording	-	Enables recording.
STOPrecording	-	Disables recording.

12.5 Collecting DR Messages at Device Startup

In some cases, especially for PSTN and SCTP, debug recording from device startup is required. The configuration for capturing DR at startup uses the same individual DR commands as described in the previous sections. However, instead of configuring each DR command separately, the commands are configured under one ini file parameter – *initialshellcommand* - which then needs to be loaded to the device.

The syntax for configuring DR commands using the *initialshellcommand* ini file parameter is as follows:

- The entire string value of DR commands included in the parameter must be enclosed in single apostrophe ('…').
- Each DR command must begin with an asterisk followed by a forward slash (***/**), and end with a semicolon (;).

For example, to send debug recording traces of all the sent and received TCP packets to a PC from device startup, set the *initialshellcommand* parameter as shown below. (These traces are sent to a PC with IP address 10.33.2.29.)

```
initialshellcommand = '*/AddIpTarget 10.33.2.29 ; */AddIPTraceRule
Host2Net TCP A A All All; */AddIPTraceRule Net2Host TCP A A All
All; */STARTrecording'
```

AudioCodes

SIP CPE Product Reference Manual

www.audiocodes.com