1 Introduction ......................................................................................................... 9

2 Deployment........................................................................................................ 11
   2.1 Operational Environment ............................................................................. 11
   2.2 Network Topology ..................................................................................... 11
   2.3 Installation Prerequisites ........................................................................... 12
      2.3.1 Installation Prerequisites for Amazon Web Services (AWS) Environment ....... 12
      2.3.1.1 IAM Role for Stack Manager ............................................................ 12
      2.3.1.2 Subnet and Elastic IP Addresses ....................................................... 13
      2.3.2 Installation Prerequisites for Microsoft Azure Environment .................... 13
      2.3.2.1 Subnet and Public IP Addresses ......................................................... 13
      2.3.3 Installation Prerequisites for Google Cloud Environment ......................... 13
      2.3.3.1 Subnet and External IP Addresses ...................................................... 13
      2.3.4 Installation Prerequisites for OpenStack Environment ............................. 14
      2.3.4.1 Provider Versus Self-Service Networks ............................................. 14
      2.3.4.2 Subnet and Floating IP Addresses ..................................................... 14
   2.4 Installation .................................................................................................... 15
      2.4.1 Creating Amazon Web Services (AWS) Instance .................................... 15
      2.4.2 Deploying Stack Manager on Microsoft Azure ......................................... 19
      2.4.3 Creating Google Cloud Virtual Machine ................................................. 24
      2.4.4 Creating OpenStack Instance ............................................................... 26
      2.4.5 Installing Stack Manager Application .................................................... 29
   2.5 Accessing the Web Interface ......................................................................... 30
   2.6 Accessing the CLI ....................................................................................... 31
   2.7 Upgrading Stack Manager .......................................................................... 31
   2.8 Post-installation Configuration ................................................................... 32
      2.8.1 Post-installation Configuration on Amazon Web Services (AWS) ................ 32
      2.8.1.1 Enabling Access to AWS API via IAM Role (Recommended Method) .... 32
      2.8.1.2 Enabling Access to AWS API via AWS Access Key (Alternative Method) 32
      2.8.2 Post-Installation Configuration on Microsoft Azure ................................. 33
      2.8.2.1 Configuring the Azure Subscription ID .............................................. 33
      2.8.2.2 Enabling Access to Azure APIs via Managed Service Identity (Recommended Method) ................................................................. 34
      2.8.2.3 Enabling Access to Azure APIs via Service Principal (Alternative Method) 38
      2.8.3 Post-Installation Configuration on Google Cloud .................................... 39
      2.8.3.1 Configuring Google Project ID ......................................................... 39
      2.8.3.2 Enabling APIs in Project .................................................................. 40
      2.8.3.3 Creating a Service Account ............................................................... 40
      2.8.3.4 Enabling Access to Google Cloud APIs via Service Account (Recommended Method) ................................................................. 40
      2.8.3.5 Enabling Access to Google Cloud APIs via Configuration File (Alternative Method) ................................................................. 41
      2.8.4 Post-installation Configuration on OpenStack ......................................... 42
      2.8.5 Verifying Configuration ......................................................................... 43
   2.9 Runtime Data .............................................................................................. 44
      2.9.1 Storing Runtime Data on AWS S3 ......................................................... 44
      2.9.2 Storing Runtime Data on Azure Storage Service .................................... 46
      2.9.3 Storing Runtime Data on Google Cloud Storage Service ........................ 46
2.9.4 Storing Runtime Data on OpenStack Object Storage Service ..................................... 47
2.9.5 Migrating Runtime Data from Local Disk to Storage Service ..................................... 47
2.10 Resource Naming ........................................................................................................... 48

3 Web Interface ...................................................................................................................... 49

3.1 Accessing the Web Interface ............................................................................................ 49
3.2 Global Configuration .......................................................................................................... 50
3.3 Creating a New Stack ......................................................................................................... 51
  3.3.1 Creating Mediant CE in Amazon Web Services (AWS) Environment ....................... 52
  3.3.1.1 Troubleshooting ........................................................................................................ 54
  3.3.2 Creating Mediant CE in Azure Environment ............................................................... 55
  3.3.2.1 Troubleshooting ........................................................................................................ 57
  3.3.3 Creating Mediant CE in Google Cloud Environment .................................................. 58
  3.3.4 Creating Mediant CE in OpenStack Environment ....................................................... 60
  3.3.5 Advanced Configuration ............................................................................................ 62
  3.3.5.1 Advanced Configuration for Mediant CE ................................................................. 63
3.4 Checking Stack State and Configuration ........................................................................... 67
3.5 Performing Operations on Stack ....................................................................................... 67
3.6 Scaling Mediant CE Stack ................................................................................................. 68
  3.6.1 Scale Out Operation ..................................................................................................... 68
  3.6.2 Scale In Operation ....................................................................................................... 68
  3.6.3 Scale To Operation ..................................................................................................... 69
3.7 Automatic Scaling .............................................................................................................. 69
  3.7.1 Cool Down Period ....................................................................................................... 70
  3.7.2 Auto Scale Step .......................................................................................................... 70
  3.7.3 Changing Cluster Size at Specific Time of Day .......................................................... 70
3.8 Modifying Stack Configuration ........................................................................................... 71
  3.8.1 Update Operation ........................................................................................................ 72
  3.8.2 Modifiable Parameters for Mediant CE ....................................................................... 73
  3.8.3 Service Interruption During Stack Update ................................................................... 74
3.9 Stopping and Starting Stack ............................................................................................... 74
3.10 Healing Stack .................................................................................................................... 75
  3.10.1 Automatic Healing ................................................................................................... 75
3.11 Deleting Stack .................................................................................................................. 76
3.12 Upgrading Software on Idle Media Components ............................................................. 76
3.13 Stack Deployment Details ............................................................................................... 76
  3.13.1 Use of Native Cloud Orchestration ............................................................................ 76
  3.13.2 Adjusting Configuration of Security Groups ............................................................. 77
  3.13.3 Using Pre-Defined Public IP Addresses ................................................................. 78
  3.13.4 Using Pre-Defined Private IP Addresses .................................................................. 78

4 CLI Interface .......................................................................................................................... 81

4.1 Accessing CLI Interface .................................................................................................... 81
4.2 Invocation ......................................................................................................................... 81
4.3 Usage Information ............................................................................................................. 81
4.4 Global Configuration ....................................................................................................... 82
4.5 Listing Available Stacks ................................................................................................. 83
4.6 Creating a New Stack ....................................................................................................... 84
  4.6.1 Creating Stack Configuration File via SBC Cluster Configuration Tool
       (Recommended Method) ................................................................................................. 84
  4.6.2 Creating Stack Configuration File Manually (Alternative Method) ......................... 90
    4.6.2.1 Sample Configuration File ...................................................................................... 91
  4.6.3 Creating a New Stack ................................................................................................. 96
4.7 Checking Stack State and Configuration .............................................................. 97
  4.7.1 Checking Idle Media Components................................................................. 99
4.8 Scaling Mediant CE Stack .................................................................................. 100
  4.8.1 Scale Out Operation ..................................................................................... 100
  4.8.2 Scale In Operation ....................................................................................... 100
  4.8.3 Scale To Operation ....................................................................................... 101
4.9 Modifying Stack Configuration ......................................................................... 102
  4.9.1 Update Operation ......................................................................................... 104
4.10 Stopping and Starting the Stack ..................................................................... 105
  4.10.1 Stopping Stack ............................................................................................ 105
  4.10.2 Starting Stack ............................................................................................. 105
4.11 Deleting Stack ................................................................................................ 106
  4.11.1 Purging Deleted Stack ............................................................................... 106
4.12 Healing Stack .................................................................................................. 107
4.13 Multiple Operations ....................................................................................... 108

5 REST API ........................................................................................................... 109
  5.1 Overview ......................................................................................................... 109
  5.2 Asynchronous Tasks ...................................................................................... 109
  5.3 Authentication.................................................................................................. 110
  5.4 Discovery .......................................................................................................... 111
  5.5 Global Configuration ....................................................................................... 111
    5.5.1 Updating Global Configuration .................................................................... 112
  5.6 Listing Available Stacks .................................................................................. 112
  5.7 Creating New Stack ......................................................................................... 113
  5.8 Checking Stack State and Configuration ......................................................... 114
  5.9 Scaling Mediant CE Stack ............................................................................... 118
    5.9.1 Scale Out Operation .................................................................................... 118
    5.9.2 Scale In Operation ...................................................................................... 118
    5.9.3 Scale To Operation ...................................................................................... 119
  5.10 Modifying Stack Configuration ..................................................................... 120
    5.10.1 Update Operation ...................................................................................... 121
  5.11 Stopping and Starting Stack .......................................................................... 122
    5.11.1 Stopping Stack .......................................................................................... 122
    5.11.2 Starting Stack ............................................................................................ 122
  5.12 Deleting Stack ............................................................................................... 123
    5.12.1 Purging Deleted Stack ............................................................................... 123
  5.13 Healing Stack ................................................................................................ 124

6 Operational Logs ............................................................................................... 125
This page is intentionally left blank.
Notice

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. Updates to this document can be downloaded from https://www.audiocodes.com/library/technical-documents.

This document is subject to change without notice.

Date Published: February-20-2020

WEEE EU Directive

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

Customer Support

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

Stay in the Loop with AudioCodes

Documentation Feedback

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

Abbreviations and Terminology

Each abbreviation, unless widely used, is spelled out in full when first used.
### Document Revision Record

<table>
<thead>
<tr>
<th>LTRT</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>28906</td>
<td>Microsoft Azure added.</td>
</tr>
<tr>
<td>28907</td>
<td>OpenStack added.</td>
</tr>
<tr>
<td>28908</td>
<td>Parameters removed: [--mc-profile {forwarding,transcoding}] and [--mc-max-pps-limit MC_MAX_PPS_LIMIT]</td>
</tr>
<tr>
<td>28909</td>
<td>Typos; Managed Service Identity section update; Upgrading Stack Manager section added.</td>
</tr>
<tr>
<td>28911</td>
<td>Google Cloud deployment added</td>
</tr>
<tr>
<td>28912</td>
<td>Code typo in Section Post-Installation Configuration on Microsoft Azure.</td>
</tr>
<tr>
<td>28913</td>
<td>OpenStack supported; majority of document update.</td>
</tr>
<tr>
<td>28914</td>
<td>Azure zones creation update; new parameters sc_num_of_interfaces, mc_num_of_interfaces, and storage_account_type.</td>
</tr>
<tr>
<td>28922</td>
<td>New subsections under Enabling Access to Azure APIs via Managed Service Identity; 'resource_group' parameter added</td>
</tr>
</tbody>
</table>
1 Introduction

Stack Manager is used for managing 'software stacks' deployed in virtual environments. It implements the complete stack lifecycle, including:

- Stack deployment
- Stack termination
- Manual stack size adjustment – using user-initiated scale-in / scale-out
- Automatic stack size adjustment – using automatic scaling
- Stack configuration update

Current implementation supports Mediant CE (Cloud Edition) and Mediant VE (Virtual Edition) SBC in the following environments:

- Amazon Web Services (AWS)
- Microsoft Azure
- Google Cloud
- OpenStack

Stack Manager implements VNFM (Virtual Network Function Manager) functionality as defined in the NFV Management and Organization (MANO) architectural framework. The following management interfaces are provided:

- Web interface
- Command line interface (CLI)
- REST API
This page is intentionally left blank.
2 Deployment

2.1 Operational Environment

Stack Manager is mostly written in Python and may be installed on one of the following operating systems:

- Ubuntu Linux versions 16.04 and 18.04
- Amazon Linux versions 1 and 2
- Red Hat Linux versions 7 and 8
- CentOS Linux versions 7 and 8
- Debian Linux Version 9

2.2 Network Topology

Stack Manager needs to have access to the following APIs for correct operation:

- Virtual Infrastructure Management API (e.g., AWS API) for deploying stack components and managing their lifecycle.
- Management API of the deployed stack (e.g., REST API of Mediant CE) for assessing operational status of deployed stack instances and managing their configuration and state.

Figure 2-1: Stack Manager Deployment Topology
2.3 Installation Prerequisites

2.3.1 Installation Prerequisites for Amazon Web Services (AWS) Environment

Prior to installing Stack Manager in the Amazon Web Services (AWS) environment, make sure that you meet the following prerequisites:

- You have an AWS account. If you don't have one, you can sign up for one on Amazon's website at http://aws.amazon.com/.
- You have created IAM Role that enables Stack Manager to access all needed AWS APIs. For more information, see Section 2.3.1.1.
- Security groups of the "Main Subnet", where Stack Manager will be deployed, allow Stack Manager to communicate with both the AWS API and the deployed Mediant CE stack instances, using the HTTPS protocol (Port 443).

2.3.1.1 IAM Role for Stack Manager

The following IAM role ensures that Stack Manager can access all needed AWS APIs for successful stack deployment and management. This role must be attached to the Stack Manager's virtual instances, as described in Section 2.4.

```json
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Action": ["ec2:*", "cloudwatch:*", "cloudformation:*", "iam:PassRole", "iam:ListInstanceProfiles"],
      "Effect": "Allow",
      "Resource": "*"
    }
  ]
}
```

- To create an IAM Role
  1. Open the AWS IAM console.
  2. Under Policies, create a new policy as specified above.
  3. Under Rules, create a new rule based on the policy created in the previous step.
2.3.1.2 Subnet and Elastic IP Addresses

Stack Manager uses the following IP addresses when communicating with Mediant CE stack instances that it deploys:

- If the stack instance has an Elastic IP assigned to its management (OAM) interface, Stack Manager uses this Elastic IP to access the stack instance’s automation and management API.
- Otherwise, Stack Manager uses the private IP address of the stack's management (OAM) interface.

To enable Stack Manager’s access to the deployed Mediant CE stack’s management APIs, it is recommended to deploy Stack Manager to the same "Main Subnet" that is used for carrying management traffic of the deployed Mediant CE stack(s).

You should always assign an Elastic IP to Stack Manager because it is needed for proper communication with the AWS API.

2.3.2 Installation Prerequisites for Microsoft Azure Environment

Prior to installing Stack Manager in the Microsoft Azure environment, make sure that you meet the following prerequisites:

- You have an Azure account. If you don't have one, you can sign up for one on Microsoft's website at http://azure.microsoft.com.
- Security groups of the "Main Subnet", where Stack Manager will be deployed, allow Stack Manager to communicate with both the Azure API and the deployed Mediant CE stack instances, using the HTTPS protocol (Port 443).

2.3.2.1 Subnet and Public IP Addresses

Stack Manager uses Public IP addresses when communicating with Mediant CE stack instances that it deploys. Therefore, it may be deployed in any subnet as long as it’s assigned with a Public IP address and is allowed to communicate with Mediant CE instances.

Nevertheless, to simplify network topology, it is recommended to deploy Stack Manager to the same "Main Subnet" used for carrying management traffic of the deployed Mediant CE stack(s).

2.3.3 Installation Prerequisites for Google Cloud Environment

Prior to installing Stack Manager in the Google Cloud environment, make sure that you meet the following prerequisites:

- You have a Google Cloud account. If you don't have one, you can sign up for one on Google’s website at http://cloud.google.com.
- Firewall Rules of the "Main Subnet", where Stack Manager will be deployed, allow Stack Manager to communicate with both the Google Cloud API and the deployed Mediant CE stack instances, using the HTTPS protocol (Port 443).

2.3.3.1 Subnet and External IP Addresses

Stack Manager uses External IP addresses when communicating with Mediant CE stack instances that it deploys. Therefore, it may be deployed in any subnet as long as it’s assigned with an External IP and is allowed to communicate with Mediant CE instances.

Nevertheless, to simplify network topology, it is recommended to deploy Stack Manager to the same "Main Subnet" that is used for carrying management traffic of the deployed Mediant CE stack(s).
2.3.4 Installation Prerequisites for OpenStack Environment

Prior to installing Stack Manager in the OpenStack environment, make sure that you meet the following prerequisites:

- The OpenStack environment contains the following components:
  - Nova
  - Neutron
  - Cinder
  - Glance
  - Heat

- Security groups of the "Main Subnet", where Stack Manager will be deployed, allow Stack Manager to communicate with both the OpenStack API and the deployed Mediant CE stack instances, using the HTTPS protocol (Port 443).

2.3.4.1 Provider Versus Self-Service Networks

Stack Manager supports deployment both in provider (flat) and self-service networks.

2.3.4.2 Subnet and Floating IP Addresses

Stack Manager uses the following IP addresses when communicating with Mediant VE/CE stack instances that it deploys:

- If the stack instance has a Floating IP address assigned to its management (OAM) interface, Stack Manager uses this Floating IP address to access the stack instance’s automation and management API.
- Otherwise, Stack Manager uses the private IP address of the stack’s management (OAM) interface.

To enable Stack Manager’s access to the deployed Mediant VE/CE stack’s management APIs, it is recommended to deploy Stack Manager to the same "Main Subnet" that is used for carrying management traffic of the deployed Mediant VE/CE stack(s). In addition, Stack Manager needs to communicate with OpenStack automation APIs. Make sure that your network topology enables such communication.
2.4 Installation

For Microsoft Azure, Stack Manager is available in the Azure Marketplace. Therefore, its deployment consists of a single step – deploying a new Virtual Machine from the Azure Marketplace. See Section 2.4.2 for detailed instructions. For other cloud environments, Stack Manager installation consists of two steps:

1. Creating the Instance / Virtual Machine: This step differs, depending on the virtual environment. For detailed instructions, see the following sections:
   - Section 2.4.1, Creating Amazon Web Services (AWS) Instance
   - Section 2.4.3, Creating Google Cloud Virtual Machine
   - Section 2.4.4, Creating OpenStack Instance

2. Installing the Stack Manager application: For detailed instructions, see Section 2.4.5, Installing Stack Manager Application

2.4.1 Creating Amazon Web Services (AWS) Instance

The following procedure describes how to create a new AWS instance for running the Stack Manager application.

➢ To create a new AWS instance for running Stack Manager application:

2. In the Instances screen, click Launch Instance.
3. Choose one of the supported operating systems (e.g., "Ubuntu Server 18.04 LTS (HVM), SSD Volume Type"), and then click Select.

Figure 2-2: Choose an Amazon Machine Image (AMI) – Step 1
4. In the Choose an Instance Type screen, choose the "t2.small" instance type, and then click **Next**; the Configure Instance Details screen appears.

**Figure 2-3: Choose an Instance Type – Step 2**

5. In the Configure Instance Details screen, configure the following:
   - **'Subnet':** Choose the "Main Subnet" that is used for connecting to the management interface of the deployed Mediant VE/CE stack(s).
   - **'Auto-assign Public IP':** Choose **Enable**.
   - **'IAM Role':** Choose the IAM role that you created for Stack Manager in Section 2.3.1.1, IAM Role for Stack Manager.

**Figure 2-4: Configure Instance Details – Step 3**
1. Click **Next**; the Add Storage screen appears.
2. Click **Next**; the Add Tags screen appears.
3. Add a Name tag to the instance, and then click **Next**; the Configure Security Group page appears.
4. Create a new or choose an existing security group that enables the following ports and protocols to communicate with the Stack Manager instance:

<table>
<thead>
<tr>
<th>Port</th>
<th>Protocol</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td>22</td>
<td>TCP</td>
<td>SSH connection to Stack Manager’s CLI interface.</td>
</tr>
<tr>
<td>80</td>
<td>TCP</td>
<td>HTTP connection to Stack Manager’s Web interface.</td>
</tr>
<tr>
<td>443</td>
<td>TCP</td>
<td>HTTPS connection to Stack Manager’s Web interface.</td>
</tr>
</tbody>
</table>

5. Click **Review and Launch**; the Review Instance Launch screen appears.
6. Click **Launch**; the Select an existing key pair … screen appears.
7. Choose an existing key pair or create a new one. Make sure that you have private key that matches the selected pair because you will need it to connect the deployed instance through the SSH protocol.

8. Click **Launch Instances**.
9. Wait until the instance is successfully launched.

10. Connect to the instance through SSH using the default username and configured SSH key. The default username depends on the image:

<table>
<thead>
<tr>
<th>Image</th>
<th>Default username</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ubuntu 16.04 and 18.04</td>
<td>ubuntu</td>
</tr>
<tr>
<td>Amazon Linux, Amazon Linux 2, RHEL 7 and 8</td>
<td>ec2-user</td>
</tr>
<tr>
<td>CentOS 7 and 8</td>
<td>Centos</td>
</tr>
</tbody>
</table>

11. By default, new AWS instances are assigned with a Public IP address that changes when the instance is stopped or started. If you want Stack Manager's Public IP address to remain unchanged, create an Elastic IP and attach it to the instance.

12. Continue with Stack Manager installation, as described in Section 2.4.5, Installing Stack Manager Application.
### 2.4.2 Deploying Stack Manager on Microsoft Azure

Stack Manager is available in Microsoft Azure Marketplace. Therefore, it is recommended that you deploy it from there, instead of manually creating a Virtual Machine and installing Stack Manager application on it.

**To deploy Stack Manager on Microsoft Azure:**

2. Navigate to Azure Marketplace (All services > Marketplace).

![Figure 2-7: Azure Marketplace](image)

4. Click the "Mediant CE Session Border Controller (SBC)" product; the Mediant CE Product overview screen appears.

![Figure 2-8: Mediant CE SBC Product Offer](image)

5. Click **Create**; a configuration wizard starts with the Basics page (Step 1).
6. In the **Basics** step, do the following:

![Figure 2-9: Basics – Step 1](image)

- **a.** In the 'Virtual Machine name' field, enter a unique name for the new virtual machine.
- **b.** In the 'Username' field, enter a username. In the 'Authentication type' field, choose an appropriate authentication type, and then enter the 'Password' or 'SSH public key' accordingly. These credentials are used to connect to the deployed Stack Manager’s CLI interface through SSH.

**Note:** Azure imposes some limitations on the username and password. For example, it prohibits the use of "Admin" for the username and requires the use of strong passwords that meet the following policy:

- A minimum of 12 characters.
- Use of three out of four of the following: lowercase characters, uppercase characters, numbers, and symbols.

- **c.** From the 'Subscription' drop-down list, select a proper subscription for your deployment.
- **d.** Under 'Resource group', click **Create new**, and then enter a new Resource Group name for your deployment.
- **e.** From the 'Location' drop-down list, select a proper location for your deployment.
- **f.** Click **OK**; the Virtual Machine Settings page (Step 2) appears.
7. In the Virtual Machine Settings step, do the following:

![Virtual Machine Settings – Step 2](image)

- **a.** Choose the Virtual machine size. Standard_B1ms instance is recommended for most deployments.
- **b.** Choose the virtual network where Stack Manager will be deployed. Specify the same network where you intend to deploy the Mediant VE/CE stack(s).
- **c.** Configure the subnet that Stack Manager will be connected to. Specify the same subnet that will be used for carrying management traffic for the deployed Mediant VE/CE stack(s).
- **d.** Configure a Public IP address to use Standard SKU:

![Virtual Machine Settings Step – Creating Public IP Address](image)

- **e.** Click OK.; the Summary page (Step 3) appears.
8. In the Summary step, review your virtual machine configuration.

**Figure 2-12: Summary – Step 3**

![Summary step 3 image]

8. In the Summary step, review your virtual machine configuration.

9. Click **OK**; the Buy page (Step 4) appears.

10. Review the Mediant CE SBC terms of use.

**Figure 2-13: Buy – Step 4**

![Buy step 4 image]

11. Click **Create** to start the virtual machine deployment.

12. Wait until the virtual machine deployment is complete, and then open the Virtual Machines screen (**All services > Virtual Machines**).

13. Select the Stack Manager virtual machine.
14. In the Overview screen, view the public IP address assigned to it.

**Figure 2-14: Determining Public IP Address**

15. In the Networking screen, verify that the following ports are open for inbound traffic:

<table>
<thead>
<tr>
<th>Port</th>
<th>Protocol</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td>22</td>
<td>TCP</td>
<td>SSH connection to Stack Manager’s CLI interface.</td>
</tr>
<tr>
<td>80</td>
<td>TCP</td>
<td>HTTP connection to Stack Manager’s Web interface.</td>
</tr>
<tr>
<td>443</td>
<td>TCP</td>
<td>HTTPS connection to Stack Manager’s Web interface.</td>
</tr>
</tbody>
</table>

16. If any port is missing, click **Add inbound port rule** and then add the port.

**Figure 2-15: Checking Inbound Port Rules**

17. Continue with post-installation configuration, as described in Section 2.8.2, Post-Installation Configuration on Microsoft Azure.
2.4.3 Creating Google Cloud Virtual Machine

The following procedure describes how to create a new Google Cloud virtual machine (VM) for running the Stack Manager application.

➢ To create a new Google Cloud virtual machine for running Stack Manager application:

2. On the VM Instances page, click Create Instance.
3. In the 'Name' field, enter a unique name for the new virtual machine.
4. Choose the Region and Zone where Stack Manager will be deployed.
5. Under the 'Machine Type' group, choose g1-small (1 shared vCPU, 1.7 GB memory).
6. Under the 'Boot disk' group, choose Ubuntu 18.04 LTS or any other supported operating system.
7. Under the 'Firewall' group, select the Allow HTTP traffic and Allow HTTPS traffic check boxes.
8. Click Management, security, disks, networking, sole tenancy.
9. In the Networking tab for the 'Network interface', choose the "Main Network" for connecting to the management interface of the deployed Mediant VE/CE stack(s).
10. If you want to be able to connect to Stack Manager's CLI interface through a regular SSH client (and not through the Google Cloud dashboard), configure the SSH keys under the Security tab. Note that the username is provided as the last part of the encoded key. For example, in the following SSH key, "admin" is the username:
    ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAA...0Sknr admin
11. Click Create.
12. By default, new Google Cloud virtual machines are assigned with ephemeral External IP addresses that change when the instance is stopped or started. If you wish Stack Manager’s External IP address to remain unchanged, allocate an External IP address and attach it to the virtual machine.

13. Continue with Stack Manager installation, as described in Section 2.4.5, Installing Stack Manager Application.
2.4.4 Creating OpenStack Instance

The following procedure describes how to create a new OpenStack instance for running the Stack Manager application.

➢ **To create an OpenStack instance for running Stack Manager application:**

1. Open the OpenStack dashboard.
2. On the Instances page, click **Launch Instance**; the Launch Instance wizard starts with the Details page.
3. In the 'Instance Name' field, enter a unique name for the new instance.

![Figure 2-17: Launch Instance Wizard - Details Page](image)

4. Click **Next**; the Source wizard page appears.
5. Select one of the supported operating system images (e.g., Ubuntu 18.04).

![Figure 2-18: Launch Instance Wizard - Source Page](image)

6. Click **Next**; the Flavor wizard page appears.
7. Select the flavor that provides 1 vCPU and 2 GB of RAM.

Figure 2-19: Launch Instance Wizard - Flavor Page

8. Click Next; the Networks wizard page appears.

9. Select the "Main Network" that will be used for connecting to the management interface of the deployed Mediant VE/CE stack(s).

Figure 2-20: Launch Instance Wizard - Networks Page

10. Click Next; the Network Ports wizard page appears.

11. Click Next; the Security Groups wizard page appears.

12. Select a security group that enables the following ports and protocols to communicate with the Stack Manager instance:

<table>
<thead>
<tr>
<th>Port</th>
<th>Protocol</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td>22</td>
<td>TCP</td>
<td>SSH connection to Stack Manager’s CLI interface.</td>
</tr>
<tr>
<td>80</td>
<td>TCP</td>
<td>HTTP connection to Stack Manager's Web interface.</td>
</tr>
<tr>
<td>443</td>
<td>TCP</td>
<td>HTTPS connection to Stack Manager's Web interface.</td>
</tr>
</tbody>
</table>
13. Click **Next**; the Key Pair wizard page appears.

Select an existing key pair or create a new one. Make sure that you have private key that matches the selected pair because you will need it to connect the deployed instance through SSH.

**Figure 2-22: Launch Instance Wizard - Key Pair Page**

14. Click **Launch Instance**.

15. Continue with Stack Manager installation, as described in Section 2.4.5, Installing Stack Manager Application.
2.4.5 Installing Stack Manager Application

The following procedure describes how to install the Stack Manager application after successfully creating the instance / virtual machine.

Note: This step is not needed if you are deploying Stack Manager from Azure Marketplace.

➢ To install Stack Manager application:

1. Log in to the launched virtual instance / machine through SSH, using the credentials obtained during the launch.

2. Run the following command to download the latest installation package:

   ```bash
   $ curl http://redirect.audiocodes.com/install/stack_mgr/stack_mgr.zip --output stack_mgr.zip
   ```

   Alternatively, you may download the installation package manually from http://redirect.audiocodes.com/install/index.html and then transfer it to the virtual instance / machine through an SCP/SFTP client (e.g., WinSCP).

3. Run the following commands to start the installation:

   ```bash
   $ unzip stack_mgr.zip
   $ sudo bash stack_mgr/install.sh
   ```

4. Continue with post-installation configuration, as described in Section 2.8, Post-installation Configuration.
2.5 Accessing the Web Interface

Stack Manager’s Web interface is accessed by connecting to the virtual machine through HTTP/HTTPS, using one of the supported web browsers:

- Google Chrome
- Firefox
- Microsoft Edge

Figure 2-23: Web Interface of Stack Manager

The default login credentials of the Web Interface are:

- Username: Admin
- Password: Admin

It is recommended to change the login credentials on first login.

➢ To change default Web credentials:

1. Log in to the Web interface.
2. Open the Configuration page.
3. In the 'Admin Username' field, enter the new username.
4. In the 'Admin Password' field, enter the new password.

Figure 2-24: Changing Web Login Credentials

5. Click Update.
2.6 Accessing the CLI

Stack Manager’s CLI interface is accessed by switching to the stack_mgr user, using the following command:

```
$ stack_mgr_cli
```

If the above command doesn’t function, close the current SSH session and then open a new one. If the problem persists, use the following alternative syntax:

```
$ sudo su - stack_mgr
```

2.7 Upgrading Stack Manager

To upgrade the Stack Manager application to the latest version, log in to the virtual instance / machine through SSH as a regular user (e.g. `ubuntu`), and then run the following command:

```
$ sudo /opt/stack_mgr/update.sh
```

Alternatively, you can upgrade Stack Manager by installing a new version using the regular installation procedure (see Section 2.4.5, Installing Stack Manager Application for details). All existing configuration and stacks will be preserved.
2.8 Post-installation Configuration

The following procedures describe post-installation configuration that ensures that Stack Manager is able to properly access cloud / virtual infrastructure APIs. The instructions depend on the cloud / virtual environment. After performing the configuration, verify that Stack Manager is able to operate normally, as described in Section 2.8.5, Verifying Configuration. For production environments, it is also recommended to configure Stack Manager to store its run-time data on cloud storage services, as described in Section 2.9, Runtime Data.

Note: The instructions described in this section use the Web interface to configure Stack Manager. The same tasks may be performed through CLI, using the configure command, as described in Section Global Configuration.

2.8.1 Post-installation Configuration on Amazon Web Services (AWS)

The following procedure describes post-installation configuration of the Stack Manager application in the Amazon Web Services (AWS) environment, which consists of the following step:

- Enabling Stack Manager virtual machine access to AWS APIs

2.8.1.1 Enabling Access to AWS API via IAM Role (Recommended Method)

Before using Stack Manager, you need to ensure that it has access to the AWS API. The recommended method for achieving this is to create an IAM role, as described in Section 2.3.1.1, IAM Role for Stack Manager, and then to attach it to the Stack Manager’s virtual instance during its creation, as described in Section 2.4.1, Creating Amazon Web Services (AWS) Instance.

2.8.1.2 Enabling Access to AWS API via AWS Access Key (Alternative Method)

This section describes an alternative method for enabling Stack Manager access to AWS APIs. For typical deployments, please use the recommended method instead, as described in Section 2.8.1.1, Enabling Access to AWS API via IAM Role (Recommended Method).

➢ To configure Stack Manager access to AWS API using access key:

1. Obtain the AWS access key. For more information on how to do this, refer to AWS documentation at https://docs.aws.amazon.com/general/latest/gr/aws-sec-cred-types.html#access-keys-and-secret-access-keys.
2. Log in to the Stack Manager Web interface.
3. Open the Configuration page.
4. Enter the access key values in the 'AWS Access Key' and 'AWS Secret Key' fields.
5. Click Update.
2.8.2 Post-Installation Configuration on Microsoft Azure

The following procedure describes post-installation configuration of the Stack Manager application in Microsoft Azure environment, which includes the following steps:

1. Configuring the Azure Subscription ID.
2. Enabling Stack Manager virtual machine access to Azure APIs.

2.8.2.1 Configuring the Azure Subscription ID

After installing Stack Manager, you need to configure the Subscription ID where it will operate.

To configure Azure Subscription ID:

2. Navigate to Subscriptions (All services > Subscriptions).
3. Locate your Azure Subscription ID.

4. Log in to the Stack Manager Web interface.
5. Open the Configuration page.
6. Enter the Azure subscription ID in the 'Azure Subscription ID' field.
2.8.2.2 Enabling Access to Azure APIs via Managed Service Identity (Recommended Method)

Before using Stack Manager, you need to ensure that it has access to Azure APIs. This section describes the recommended method for achieving this through the Managed Service Identity. The method consist of two steps:

1. Enabling Managed Service Identity for the Stack Manager virtual machine.
2. Assigning a proper IAM role to the Stack Manager virtual machine.

An alternative method is to use the service principal, as described in Section 2.8.2.3, Enabling Access to Azure APIs via Service Principal (Alternative Method).

Managed Service Identity (MSI) enables the assignment of access control (IAM) roles to a specific Azure virtual machine deployed in Azure.

➢ To enable Managed Service Identity:
   2. Navigate to the Virtual Machines page.
   3. Select the Stack Manager virtual machine.
4. In the Navigation menu, click **Identity**, and then enable Managed Service Identity.

**Figure 2-27: Configuring Virtual Machine’s Managed Service Identity**

Once you have performed the above procedure, you should grant the Stack Manager virtual machine permissions to access all needed Azure APIs for successful stack deployment and management. There are several ways to achieve this:

- **Option 1 (recommended):** Assign Stack Manager with the "Contributor" role at the Subscription level.
- **Option 2:** Assign Stack Manager with custom IAM roles at Subscription, Network and Resource Group levels.

### 2.8.2.2.1 Option 1: "Contributor" Role at Subscription Level

This method provides Stack Manager with complete access to Subscription resources, including the ability to create new Resource Groups. This method is recommended for most users, as it's simple to provision and doesn't impose any restrictions on Stack Manager functionality.

- **To assign Stack Manager with "Contributor" role at Subscription level:**
  2. Navigate to the Subscriptions page.
  3. Select your subscription.
  4. In the Navigation menu, click **Access Control (IAM)**, and then click **Add a role assignment**:
     - a. From the 'Role' drop-down list, select **Contributor**.
     - b. From the 'Assign access to' drop-down list, select **Virtual Machine**.
c. From the 'Select' drop-down list, select the name of Stack Manager's virtual machine.

d. Click Save.

Figure 2-28: Adding Role Assignment

2.8.2.2 Option 2: Custom IAM Roles at Subscription, Network and Resource Group Levels

This method limits Stack Manager administrative access to the specific pre-defined Resource Group(s). It is more complicated to provision and slightly complicates stack creation. Therefore, this method is recommended for advanced users who want to minimize IAM permissions granted to the Stack Manager.

With this method, Stack Manager is assigned the following IAM roles:

<table>
<thead>
<tr>
<th>Scope</th>
<th>IAM Role</th>
</tr>
</thead>
<tbody>
<tr>
<td>Subscription</td>
<td>Custom IAM role that includes read-only access to specific resources only. This is needed for displaying &quot;Create new stack&quot; Web UI dialog and validating stack configuration during create, modify, update, and heal operations.</td>
</tr>
<tr>
<td>Virtual Network</td>
<td>Custom IAM role that grants Stack Manager the ability to deploy new virtual machines into the specific Virtual Network(s). The role is assigned only for specific Virtual Networks where new stacks will be deployed.</td>
</tr>
<tr>
<td>Resource Group</td>
<td>&quot;Contributor&quot; IAM role that grants Stack Manager full administrative access, including the ability to create, modify and delete resources. The role is assigned only for specific Resource Group(s) that must be pre-created prior to stack deployment.</td>
</tr>
</tbody>
</table>

Note: When using this method, an empty Resource Group must be manually created prior to stack deployment and Stack Manager must be assigned the "Contributor" role in it. The name of this Resource Group must be specified during new stack creation through the Advanced Config parameter resource_group.
To assign Stack Manager with custom IAM roles at Subscription, Network and Resource Group levels:

1. Create the following two custom IAM roles:

   **Custom IAM Role 'Stack Manager Subscription Role'**

   ```json
   { 
     "Name": "Stack Manager Subscription Role",
     "IsCustom": true,
     "Description": "Subscription role for AudioCodes Stack Manager.",
     "Actions": [
       "Microsoft.Network/virtualNetworks/read",
       "Microsoft.Network/virtualNetworks/subnets/read",
       "Microsoft.Network/publicIPAddresses/read",
       "Microsoft.Compute/images/read",
       "Microsoft.Compute/skus/read",
       "Microsoft.Subscription/SubscriptionDefinitions/read",
       "Microsoft.Compute/virtualMachines/vmSizes/read"
     ],
     "NotActions": [],
     "DataActions": [],
     "NotDataActions": [],
     "AssignableScopes": [
       "/subscriptions/{subscriptionId}"
     ]
   }
   ```

   **Custom IAM Role 'Stack Manager Network Role'**

   ```json
   { 
     "Name": "Stack Manager Network Role",
     "IsCustom": true,
     "Description": "Network role for AudioCodes Stack Manager.",
     "Actions": [
       "Microsoft.Network/virtualNetworks/subnets/join/action"
     ],
     "NotActions": [],
     "DataActions": [],
     "NotDataActions": [],
     "AssignableScopes": [
       "/subscriptions/{subscriptionId}"
     ]
   }
   ```

Refer to Azure documentation at [https://docs.microsoft.com/en-us/azure/role-based-access-control/custom-roles](https://docs.microsoft.com/en-us/azure/role-based-access-control/custom-roles) for detailed instructions on how to create custom IAM roles.

3. Navigate to the Subscriptions page.
4. Select your subscription.
5. In the Navigation menu, click **Access Control (IAM)**, and then click **Add a role assignment**:
   a. From the 'Role' drop-down list, select **Stack Manager Subscription Role**.
   b. From the 'Assign access to' drop-down list, select **Virtual Machine**.
   c. From the 'Select' drop-down list, select the name of Stack Manager's virtual machine.
   d. Click **Save**.
6. Navigate to the Virtual Networks page.
7. Select the network where new stacks will be deployed.
8. In the Navigation menu, click **Access Control (IAM)**, and then click **Add a role assignment**:
   a. From the 'Role' drop-down list, select **Stack Manager Network Role**.
   b. From the 'Assign access to' drop-down list, select **Virtual Machine**.
   c. From the 'Select' drop-down list, select the name of Stack Manager's virtual machine.
   d. Click **Save**.
10. Click **Add** to create a new Resource Group(s) where new stacks will be deployed. Each stack will require a dedicated Resource Group that must be empty prior to stack creation.
    a. Enter the Resource Group name.
    b. From the 'Region' drop-down list, select the region where the new stack will be deployed.
    c. Click **Create**.
11. Select the created Resource Group(s).
12. In the Navigation menu, click **Access Control (IAM)**, and then click **Add a role assignment**:
    a. From the 'Role' drop-down list, select **Contributor**.
    b. From the 'Assign access to' drop-down list, select **Virtual Machine**.
    c. From the 'Select' drop-down list, select the name of Stack Manager's virtual machine.
    d. Click **Save**.
13. Restart the Stack Manager virtual machine to apply the new IAM credentials.

### 2.8.2.3 Enabling Access to Azure APIs via Service Principal (Alternative Method)

This section describes an alternative method for enabling Stack Manager access to Azure APIs. For typical deployments, please use the recommended method instead, as described in Section 2.8.2.2, Enabling Access to Azure APIs via Managed Service Identity (Recommended Method).

➢ **To configure Stack Manager access to Azure API using Service Principal:**

1. Create an Azure Service Principal, as described in the Azure documentation at [https://docs.microsoft.com/en-us/azure/active-directory/develop/app-objects-and-service-principals](https://docs.microsoft.com/en-us/azure/active-directory/develop/app-objects-and-service-principals). Assign an appropriate IAM role(s) to the created Azure Service Principal, as described in the previous section.
2. Log in to the Stack Manager Web interface.
3. Open the Configuration page.
4. Enter the values in the 'Azure Tenant ID', 'Azure Client ID' and 'Azure Secret' fields.
5. Click **Update**.
2.8.3 Post-Installation Configuration on Google Cloud

The following procedure describes post-installation configuration of the Stack Manager application in Google Cloud environment, which includes the following steps:

1. Configuring Google Project ID.
2. Enabling Google Cloud APIs in the Project.
3. Enabling Stack Manager virtual machine access to Google Cloud APIs.

2.8.3.1 Configuring Google Project ID

After installing Stack Manager, you need to configure the Project ID where it will operate.

➢ To configure Google Project ID:

1. In Google Cloud Platform Console, go to the Home > Dashboard (https://console.cloud.google.com/home/dashboard), and then determine your project ID.

   ![Figure 2-29: Determining Google Project ID](image)

2. Log in to the Stack Manager Web interface.
3. Open the Configuration page.
4. In the 'Google Project' field, enter the Project ID.
5. Click Update.
2.8.3.2 Enabling APIs in Project

The following Google Cloud APIs must be enabled in the Project for normal Stack Manager operation:
- Compute Engine API
- Cloud Deployment Manager V2 API
- Cloud Resource Manager API

To enable APIs in the project:
1. In the Google Cloud Platform Console, go to the API & Services > Dashboard page (https://console.cloud.google.com/apis/dashboard).
2. Click Enable APIs And Services.
3. Type the API name, and then select it from the list.
4. Click Enable to enable the API.
5. Repeat the above steps for all APIs required by the Stack Manager.

2.8.3.3 Creating a Service Account

Service Accounts are used to manage application permissions.

To create a Service Account:
1. In the Google Cloud Platform Console, go to the IAM & admin > Service Accounts page (https://console.cloud.google.com/iam-admin/serviceaccounts).
2. Click Create service account.
3. Enter the service account name, for example, "stack-mgr", and provide a description.
4. Click Create to create the account.
5. On the Service account permissions (optional) page displayed immediately afterwards, assign the following IAM roles to the service account, and then click Continue.
   a. Compute Engine > Compute Admin.
   b. Deployment Manager > Deployment Manager Editor.
6. On the Grant users access to this service account (optional) page displayed immediately afterwards, click Done.
8. Verify that the service account has been successfully created and is assigned with Compute Admin and Deployment Manager Editor roles.

2.8.3.4 Enabling Access to Google Cloud APIs via Service Account (Recommended Method)

Before using Stack Manager, you need to ensure that it has access to Google Cloud API. This section describes the recommended method for achieving this through the Service Account assigned to the Stack Manager virtual machine.

An alternative method is to use the configuration file, as described in Section 2.8.3.5, Enabling Access to Google Cloud APIs via Configuration File (Alternative Method).

To assign Service Account to Stack Manager virtual machine:
1. In the Google Cloud Platform Console, go to the Compute Engine > VM Instances page (https://console.cloud.google.com/compute/instances).
2. Click the Stack Manager VM.
3. On the VM instance details page, click Edit.
4. For Service account, select the Service Account that you created in Section 2.8.3.3, Creating a Service Account.
5. Click Save.

2.8.3.5 Enabling Access to Google Cloud APIs via Configuration File (Alternative Method)

This section describes an alternative method for enabling Stack Manager access to Google Cloud APIs. For typical deployments, please use the recommended method instead, as described in Section 2.8.3.4, Enabling Access to Google Cloud APIs via Service Account (Recommended Method).

➢ To enable access to Google Cloud APIs via configuration file:
1. In the Google Cloud Platform Console, go to the IAM & admin > Service Accounts page (https://console.cloud.google.com/iam-admin/serviceaccounts).
2. Click the Service Account that you created in Section 2.8.3.3, Creating a Service Account.
3. Click Edit.
4. Click Create Key.
5. Choose the JSON key type, and then click Create.
6. The credentials file, which contains the generated key, is downloaded and saved to your computer. Move the file to a permanent location and write down its complete name and path.
7. Log in to the Stack Manager Web interface.
8. Open the Configuration page.
9. In the 'Google Credentials' field, enter the complete path to the credentials file.
10. Click Update.
Post-installation Configuration on OpenStack

The following procedure describes post-installation configuration of the Stack Manager application in the OpenStack environment.

To perform post-installation configuration of Stack Manager in OpenStack environment:

1. Obtain credentials for application access to your OpenStack installation.
2. Create the configuration file `clouds.yaml`, which will be used by Stack Manager to access OpenStack APIs. Below shows an example OpenStack configuration file:

```yaml
clouds:
  openstack-se2:
    region_name: RegionOne
    auth:
      auth_url: http://10.4.220.50:5000/v3
      username: admin
      password: 123456
      project_name: admin
      project_domain_name: Default
      user_domain_name: Default
```

Change the configuration parameters to match your OpenStack installation. Refer to the openstacksdk documentation at http://docs.openstack.org/openstacksdk for more information.

3. Place the file in one of the following locations:
   - `/var/stack_mgr/.config/openstack`
   - `/etc/openstack`
   Make sure that the file is readable by user `stack_mgr`.
4. Log in to the Stack Manager Web interface.
5. Open the Configuration page.
6. In the ‘OpenStack Cloud Name’ field, enter the value ("openstack-se2" in the example above).
7. Click Update.
2.8.5 Verifying Configuration

After completing post-installation configuration, perform the following steps to verify that Stack Manager can operate normally.

➢ To verify Stack Manager configuration:
1. Log in to the Stack Manager Web interface.
2. Open the Configuration page.
3. Click Verify.
4. Wait until the operation completes, and then check its output.

Figure 2-30: Verifying Stack Manager Configuration
2.9 Runtime Data

Stack Manager uses stack descriptors to keep information about created stacks, including their configuration and references to all corresponding resources. By default, Stack Manager stores this information on the local file system in the /opt/stack_mgr/data directory. However, you may configure Stack Manager to store the stack descriptors in the cloud storage services, namely:

- AWS Simple Cloud Storage Service (S3)
- Microsoft Azure Storage Service
- Google Cloud Storage Service
- OpenStack Object Storage Service (swift)

Doing so significantly improves runtime data availability and provides service continuity if the Stack Manager instance must be rebuilt.

Note: Stack descriptors are for internal Stack Manager use and should not be manipulated by the user.

2.9.1 Storing Runtime Data on AWS S3

The procedure below describes how to configure Stack Manager to store its runtime data on AWS S3.

➢ To configure Stack Manager to store runtime data on AWS S3:

2. Create a new S3 bucket in the same region where the Stack Manager instance is deployed. Enter the bucket name (e.g., "stack-mgr").

Figure 2-31: Create Bucket
3. Create a new IAM policy that allows the Stack Manager instance to access data in the created S3 bucket. In the 'Bucket name' field, replace stack-mgr with the actual name of the bucket that you created.

```json
```

4. Attach the created IAM policy to the Stack Manager instance (in addition to the policy created in Section 2.3.1.1, IAM Role for Stack Manager).

5. Log in to the Stack Manager Web interface.

6. Open the Configuration page.

7. In the 'AWS S3 Bucket' field, enter the value ("stack-mgr" in the example above).

8. If you want Stack Manager runtime data to be stored in some folder(s), configure the 'AWS S3 Prefix' field to some value that ends with "/" (e.g., "stack-mgr/").

9. Click Update.

10. Click Verify to verify configuration.
2.9.2 Storing Runtime Data on Azure Storage Service

The procedure below describes how to configure Stack Manager to store its runtime data on Microsoft Azure Storage Service.

➢ To configure Stack Manager to store runtime data on Azure Storage Service:

2. Navigate to the Storage Accounts page (All services > Storage Accounts).
3. Create a new Storage Account in the same location where the Stack Manager virtual machine is deployed.
4. Locate the access key for the Storage Account under the Access keys tab.
5. Go to the Blobs service, and then create a new container.
6. Log in to the Stack Manager Web interface.
7. Open the Configuration page.
8. In the 'Azure Blob Account Name', 'Azure Blob Account Key', and 'Azure Blob Container' fields, enter the values.
9. Click Update.
10. Click Verify to verify configuration.

Note: Instead of using the Access Key as described above, Stack Manager may be configured to access Azure Storage Service using a shared access signature (SAS) token. For this you need to use the 'Azure Blob SAS token' configuration parameter.

2.9.3 Storing Runtime Data on Google Cloud Storage Service

The procedure below describes how to configure Stack Manager to store its runtime data on Google Cloud Storage Service.

➢ To configure Stack Manager to store runtime data on Google Cloud Storage Service:

1. In the Google Cloud Platform Console, go to the Storage > Browser page ([https://console.cloud.google.com/storage/browser](https://console.cloud.google.com/storage/browser)).
2. Create a bucket where Stack Manager runtime data will be stored.
3. Create folder(s) inside the bucket, if needed.
4. Go to the IAM & admin > IAM page ([https://console.cloud.google.com/iam-admin/iam](https://console.cloud.google.com/iam-admin/iam)).
   Assign the following IAM role to the Stack Manager service account: Storage > Storage Object Admin.
5. Log in to the Stack Manager Web interface.
6. Open the Configuration page.
7. In the 'Google Storage Bucket' field, enter the value.
8. If you want Stack Manager runtime data to be stored in some folder(s), configure the 'Google Storage Prefix' field to some value that ends with "/" (e.g., "stack-mgr/").
9. Click Update.
10. Click Verify to verify configuration.
2.9.4 Storing Runtime Data on OpenStack Object Storage Service

The procedure below describes how to configure Stack Manager to store its runtime data on OpenStack Object Storage Service (swift).

➢ To configure Stack Manager to store runtime data on OpenStack Object Storage Service (swift):

1. Open the OpenStack dashboard.
2. Navigate to Object Store > Containers page.
3. Create a new Object Storage (swift) container.
4. Log in to the Stack Manager Web interface.
5. Open the Configuration page.
6. In the ‘Openstack Container’ field, enter the value.
7. Click Update.
8. Click Verify to verify configuration.

2.9.5 Migrating Runtime Data from Local Disk to Storage Service

If you started working with Stack Manager while it was configured to store run-time data on local disk and later decided to migrate to the cloud-specific storage service, use the following procedure to migrate the data:

1. Download all .json files from the /opt/stack_mgr/data folder to your computer.
2. Remove the .json extension from all the downloaded files.
3. Upload all the files to the proper container / folder on the storage service.
2.10 Resource Naming

By default, resources created by Stack Manager (e.g., virtual machines) use the following naming convention: <stack name>-<resource name>

For example, for stack 'stack1', the corresponding resources are named "stack1-sc-1", "stack1-mc-1" and so on.

It is possible to define additional prefixes that will be added to created resources. The prefix would typically end with a dash "-_". For example, if you configure it as "lab1-_", the corresponding resources are named "lab1-stack1-sc-1", and so on.

To configure a name prefix:

1. Log in to the Stack Manager Web interface.
2. Open the Configuration page.
3. In the 'Name Prefix' field, enter the value (e.g., "lab1-").
4. Click Update.

Note: The 'Name Prefix' field should be configured prior to any Mediant VE/CE stack creation. Do not change it if some stacks already exist.
3 Web Interface

3.1 Accessing the Web Interface

Stack Manager’s Web interface is accessed by connecting to the virtual machine through HTTP/HTTPS, using one of the supported web browsers:

- Google Chrome
- Firefox
- Microsoft Edge

![Figure 3-1: Accessing Web Interface](image)

The default login credentials of the Web interface are:

- Username: **Admin**
- Password: **Admin**

It is recommended to change the default login credentials on first login.

To change the default Web credentials:

1. Log in to the Web interface.
2. Open the Configuration page.
3. In the 'Admin Username' and 'Admin Password' fields, enter the new username and password respectively.
4. Click **Update**.
3.2 Global Configuration

The Configuration page contains global configuration parameters of the Stack Manager application. All the parameters are described in Section 2, Deployment. If you change the value of a parameter, click Update to update configuration. To verify current configuration, click Verify. See Section 2.8.5, Verifying Configuration for more information.

![Configuration Page](image)

Figure 3-2: Configuration Page
3.3 Creating a New Stack

The procedure below describes how to create a new stack.

➢ To create a new Mediant VE/CE stack:

1. Open the Stacks page.

![Figure 3-3: Creating a New Stack](image)

2. Click Create new stack; the Create new stack dialog box appears.

![Figure 3-4: Create New Stack Dialog](image)

3. In the 'Name' field, enter the stack name.

4. From the 'Environment' drop-down list, select the public cloud / virtual environment; the dialog box is updated with the relevant parameters.

5. Refer to the following sections for detailed instructions for each public cloud / virtual environment.

3.3.1 Creating Mediant CE in Amazon Web Services (AWS) Environment

The following configuration parameters should be configured (in the Create new stack dialog) for Mediant CE stack in Amazon Web Services (AWS) environment:

- 'Name': Defines the stack name, which can contain lowercase or uppercase letters, digits, and the dash symbol.
- 'Stack type': Mediant CE.
- 'Environment': AWS.
- 'Region': Defines the region where Mediant CE is to be deployed.
- 'Key Pair': Defines the key pair for logging in to the Mediant CE CLI through SSH. Alternatively, you can log in using the password specified below.
- 'IAM Role': Defines the name of the IAM role that enables Mediant CE access to AWS APIs for network reconfiguration in case of SC switchover. Refer to Mediant Cloud Edition Installation Manual for detailed instructions on how to create it.

**Networking:**

- 'VPC': Defines the Virtual Private Cloud where Mediant CE is to be deployed.
- 'Cluster Subnet': Defines the subnet within the VPC for internal communication between Mediant CE components. The subnet must have a private EC2 endpoint or NAT Gateway configured as the default route. Refer to Mediant Cloud Edition Installation Manual for detailed instructions on how to create it.
- 'Main Subnet': Defines the subnet within the VPC for carrying management traffic (e.g., connecting to the Mediant CE Web or SSH interface). The subnet can also be used for carrying signaling and media traffic.
- '1st and 2nd Additional Subnet': Defines additional subnets for carrying signaling and media traffic. If not needed, leave them as -- none --.

**Note:** All specified subnets must reside in the same Availability Zone.

**Media Components:**

- 'Profile': Defines the operational mode of MCs (forwarding or transcoding). Note that the profile must be specified during Mediant CE creation and cannot be altered afterwards.
- 'Max Number': Defines the total number of MCs that will be created. It also defines the higher boundary for scale-out operation.
- 'Min Number': Defines the number of MCs that will be initially active after Mediant CE creation. It also defines the lower boundary for scale-in operation.

**Admin User:**

- 'Username': Defines the username for logging in to the Mediant CE Web or SSH interface.
- 'Password': Defines the password for logging in to the Mediant CE Web or SSH interface.

**Additional Config:** For additional configuration parameters, see Section 3.3.5, Advanced Configuration.
Once you have configured all the above parameters, click **Create** to create the Mediant CE stack instance. The operation progress is displayed at the top of the page.

**Figure 3-6: Creating Mediant CE in AWS environment**
### 3.3.1.1 Troubleshooting

The following table lists common problems during Mediant CE stack creation in the AWS environment and their corresponding solutions.

**Table 3-1: Troubleshooting Mediant CE Stack Creation in AWS Environment**

<table>
<thead>
<tr>
<th>Problem</th>
<th>Reason</th>
<th>Solution</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mediant CE stack creation freezes at the &quot;Creating media components&quot; step for more than 10 minutes. No Media Component instances are shown in the AWS dashboard.</td>
<td>You haven't subscribed to the Mediant VE offer in AWS Marketplace.</td>
<td>Subscribe to Mediant VE offer in AWS Marketplace, as described in <em>Mediant Cloud Edition Installation Manual</em>.</td>
</tr>
<tr>
<td>The IAM role specified during Mediant CE stack creation doesn’t exist.</td>
<td></td>
<td>Create an IAM role for Mediant CE, as described in <em>Mediant Cloud Edition Installation Manual</em> and specify its name in the Mediant CE Create stack dialog box.</td>
</tr>
</tbody>
</table>
3.3.2 Creating Mediant CE in Azure Environment

The following configuration parameters should be configured (in the Create new stack dialog) for Mediant CE stack in the Azure environment:

- **'Name':** Defines the stack name, which can contain lowercase or uppercase letters, digits, and the dash symbol.
- **'Stack type':** Mediant CE.
- **'Environment':** Azure.
- **'Region':** Defines the region where Mediant CE is to be deployed.
- **'Zones':** By default, the parameter is set to "auto", which deploys Mediant CE components into a single Proximity Placement Group with two Availability Sets (each containing two fault and update domains) for signaling and media components, respectively. This deployment topology minimizes network latency between Mediant CE components while still providing adequate redundancy at the infrastructure level. If you want to deploy Mediant CE into two Availability Zones instead, you may define a comma-separated list of two zone names (e.g., "1,2"). In this scenario, Mediant CE components will be evenly spread across these two zones. Note however, that such deployment topology may suffer from intermittent network latency between zones, which may affect internal communication between Mediant CE components and cause SC/MC switchovers.

**Networking:**
- **'Virtual Network':** Defines the virtual network where Mediant CE is to be deployed.
- **'Cluster Subnet':** Defines the subnet used for internal communication between Mediant CE components.
- **'Main Subnet':** Defines the subnet for carrying management traffic (e.g., connecting to the Mediant CE Web or SSH interface). The subnet can also be used for carrying signaling and media traffic.
- **1st and 2nd Additional Subnet':** Defines additional subnets for carrying signaling and media traffic. If not needed, leave them as -- none --.

**Media Components:**
- **'Profile':** Defines the operational mode of MCs (forwarding or transcoding). Note that the profile must be specified during Mediant CE creation and cannot be altered afterwards.
- **'Max Number':** Defines the total number of MCs that will be created. It also defines the higher boundary for scale-out operation.
- **'Min Number':** Defines the number of MCs that will be initially active after Mediant CE creation. It also defines the lower boundary for scale-in operation.

**Admin User:**
- **'Username':** Defines the username for logging in to the Mediant CE Web or SSH interface.
- **'Password':** Defines the password for logging in to the Mediant CE Web or SSH interface.

**Note:** Azure imposes some limitations on the username and password. For example, it prohibits the use of "Admin" for username and requires the use of strong passwords that meet the following policy:

- A minimum of 12 characters.
- Use of three out of four of the following: lowercase characters, uppercase characters, numbers, and symbols.
### Additional Config:

- **Management ports**: Defines a comma-separated list of inbound ports and corresponding transport protocols for the management interface, for example, "22/tcp,80/tcp,443/tcp,161/udp".
- **Signaling ports**: Defines a comma-separated list of inbound ports and corresponding transport protocols for signaling interfaces, for example, "5060/udp,5060/tcp,5061/tcp".
- For additional configuration parameters, see Section 3.3.5, Advanced Configuration.

![Figure 3-7: Configuring Mediant CE in Azure Environment](image-url)
Once you have configured all the above parameters, click **Create** to create the Mediant CE stack instance. The operation progress is displayed at the top of the page.

**Figure 3-8: Creating Mediant CE in Azure Environment**

---

**Note:** If Stack Manager is assigned with custom IAM roles at Subscription, Network and Resource Group levels, as described in Section 2.8.2.2.2, Option 2: Custom IAM Roles at Subscription, Network and Resource Group Levels, an empty Resource Group must be manually created prior to stack deployment and Stack Manager must be assigned with “Contributor” role in it. The name of this Resource Group must be specified during stack creation by the Advanced Config parameter `resource_group`.

3.3.2.1 **Troubleshooting**

The following table lists common problems during Mediant CE stack creation in the Azure environment and their corresponding solutions.

**Table 3-2: Troubleshooting Mediant CE Stack Creation in Azure Environment**

<table>
<thead>
<tr>
<th>Problem</th>
<th>Reason</th>
<th>Solution</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mediant CE stack creation fails with the error message &quot;Legal terms have not been accepted for this item on this subscription&quot;.</td>
<td>You haven't subscribed to the Mediant VE offer in Azure Marketplace.</td>
<td>Subscribe to Mediant VE offer in Azure Marketplace by deploying a demo instance of it. Refer to Mediant Cloud Edition Installation Manual for detailed description.</td>
</tr>
</tbody>
</table>
3.3.3 Creating Mediant CE in Google Cloud Environment

The following configuration parameters should be configured (in the Create new stack dialog) for Mediant CE stack in Google Cloud environment:

- **'Name'**: Defines the stack name, which can contain lowercase or uppercase letters, digits, and the dash symbol.
- **'Stack type'**: Mediant CE.
- **'Environment'**: Google.
- **'Region'**: Defines the region where Mediant CE is to be deployed.
- **'Zones'**: Defines a comma-separated list of two Availability Zones within the specified Region. Mediant CE components will be evenly spread across these two zones.
- **'Image'**: Defines the name of the Mediant VE/CE image. Refer to Mediant Cloud Edition Installation Manual for detailed instructions on how to upload it to your account.

**Networking**:
- **'Cluster Subnet'**: Defines the subnet used for internal communication between Mediant CE components.
- **'Main Subnet'**: Defines the subnet for carrying management traffic (e.g. connecting to the Mediant CE Web or SSH interface) and signaling traffic. The subnet can also be used for carrying media traffic.
- **1st and 2nd Additional Subnet'**: Defines additional subnets used for carrying media traffic. If not needed leave them as -- none --.

**Media Components**:
- **'Profile'**: Defines the operational mode of MCs (forwarding or transcoding). Note that the profile must be specified during Mediant CE creation and cannot be altered afterwards.
- **'Max Number'**: Defines the total number of MCs that will be created. It also defines the higher boundary for scale-out operation.
- **'Min Number'**: Defines the number of MCs that will be initially active after Mediant CE creation. It also defines the lower boundary for scale-in operation.

**Admin User**:
- **'Username'**: Defines the username for logging in to the Mediant CE Web or SSH interface.
- **'Password'**: Defines the password for logging in to the Mediant CE Web or SSH interface.

**Additional Config**: For additional configuration parameters, see Section 3.3.5, Advanced Configuration.
Once you have configured all the above parameters, click **Create** to create the Mediant CE stack instance. The operation progress is displayed at the top of the page.

---

**Figure 3-10: Creating Mediant CE in Google Cloud Environment**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Environment</th>
<th>State</th>
<th>IP Address</th>
</tr>
</thead>
<tbody>
<tr>
<td>ce1</td>
<td>Mediant CE</td>
<td>Google</td>
<td>creating</td>
<td></td>
</tr>
</tbody>
</table>
3.3.4 Creating Mediant CE in OpenStack Environment

The following configuration parameters should be configured (in the Create new stack dialog) for Mediant CE stack in OpenStack environment:

- **'Name':** Defines the stack name, which can contain lowercase or uppercase letters, digits, and the dash symbol.
- **'Stack type':** Mediant CE.
- **'Environment':** OpenStack.
- **'Image':** Defines the name of the Mediant VE/CE image. Refer to Mediant Cloud Edition Installation Manual for detailed instructions on how to upload it to your account.
- **'Key Pair':** Defines the key pair for logging in to the Mediant CE CLI through SSH. Alternatively, you can log in using the password specified below.

**Networking:**

- **'Cluster Subnet':** Defines the subnet for internal communication between Mediant CE components.
- **'Main Subnet':** Defines the subnet for carrying management traffic (e.g., connecting to the Mediant CE Web or SSH interface). The subnet can also be used for carrying signaling and media traffic.
- **'1st and 2nd Additional Subnet':** Defines additional subnets for carrying signaling and media traffic. If not needed, leave them as -- none --.

**Signaling Components:**


**Media Components:**

- **'Profile':** Defines the operational mode of MCs (forwarding or transcoding). Note that the profile must be specified during Mediant CE creation and cannot be altered afterwards.
- **'Max Number':** Defines the total number of MCs that will be created. It also defines the higher boundary for scale-out operation.
- **'Min Number':** Defines the number of MCs that will be initially active after Mediant CE creation. It also defines the lower boundary for scale-in operation.

**Admin User:**

- **'Username':** Defines the username for logging in to the Mediant CE Web or SSH interface.
- **'Password':** Defines the password for logging in to the Mediant CE Web or SSH interface.

**Additional Config:** For additional configuration parameters, see Section 3.3.5, Advanced Configuration.
Once you have configured all the above parameters, click **Create** to create the Mediant CE stack instance. The operation progress is displayed at the top of the page.

---

**Figure 3-11: Configuring Mediant CE in OpenStack Environment**

![Create new stack form]

---

**Figure 3-12: Creating Mediant CE in OpenStack Environment**

![Creating stack 'ctl']
3.3.5 Advanced Configuration

The Create new stack dialog includes the Advanced Config group that can be used to specify advanced configuration parameters during stack creation. Specify parameters using the following format:

<parameter name> = <value>

You can specify multiple parameters on multiple lines.

Figure 3-13: Advanced Configuration Parameters
### Advanced Configuration for Mediant CE

The following table describes advanced parameters available for Mediant CE.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Applicable Environment</th>
<th>Description</th>
</tr>
</thead>
</table>
| sc_ha_mode       | All                    | Defines the number of SCs. Supported values:  
  - **enable** (default): Two SCs are created and operate in 1+1 HA mode.  
  - **disable**: One SC is created.  
  Example:  
  ```plaintext  
  sc_ha_mode = disable  
  ``` |
| sc_public_ips    | All                    | Defines the SC's network interface names for which public IP addresses are allocated. By default, public IP addresses are allocated for all interfaces, except the one connected to the Cluster Subnet. If you want to use private IP addresses on some interface(s), change this parameter accordingly, excluding the corresponding interface name.  
  Syntax: comma-separated list of interface names  
  Examples:  
  ```plaintext  
  sc_public_ips = eth1,eth3  
  sc_public_ips = eth1  
  sc_public_ips = .  
  ```  
  **Note:**  
  - In Azure, network interfaces listed in this parameter are placed behind the Public Load Balancer. The other interfaces are placed behind the Internal Load Balancer.  
  - In Google Cloud, only eth0 interface is supported and is placed behind the Network Load Balancer. However, multiple External IP addresses can be created using this special parameter syntax:  
    ```plaintext  
    sc_public_ips = eth0:<num>  
    ``` |
| mc_public_ips    | All                    | Defines the MC's network interface names for which public IP addresses are allocated. By default, public IP addresses are allocated for all interfaces, except the one connected to the Cluster Subnet. If you want to use private IP addresses on some interface(s), change this parameter accordingly, excluding the corresponding interface name.  
  Syntax: comma-separated list of interface names  
  Examples:  
  ```plaintext  
  mc_public_ips = eth1,eth3  
  mc_public_ips = eth1  
  mc_public_ips = .  
  ```
<table>
<thead>
<tr>
<th>Parameter</th>
<th>Applicable Environment</th>
<th>Description</th>
</tr>
</thead>
</table>
| sc_additional_ips  | All                    | Defines the additional private IP addresses for SCs. Syntax: comma-separated list of interface names followed by the number of corresponding additional IP addresses. Examples:  

```
sc_additional_ips = eth2:1
sc_additional_ips = eth2:1,eth3:2
```

**Note:**
- For Azure deployments with two SCs (in 1+1 HA configuration), the parameters is essentially meaningless because additional private IP addresses are not placed behind any Load Balancer.
- For Google Cloud deployments with two SCs (in 1+1 HA configuration), only eth0 interface is supported and additional private IP addresses allocated for it are implemented by the Internal Load Balancer. |
| mc_additional_ips  | All                    | Defines additional private IP addresses for MCs. Syntax: comma-separated list of interface names followed by the number of corresponding additional IP addresses. Examples:  

```
mc_additional_ips = eth2:1
mc_additional_ips = eth2:1,eth3:2
```
| sc_num_of_interfaces | All | Defines the number of network interfaces for SCs. By default, all components (SCs and MCs) are connected to all subnets specified during stack creation. If you want some subnets to be connected to MCs only, use this parameter to reduce the number of network interfaces for SCs. Example:  

```
esc_num_of_interfaces = 2
```
| mc_num_of_interfaces | All  | Defines the number of network interfaces for MCs. By default, all components (SCs and MCs) are connected to all subnets specified during stack creation. If you want some subnets to be connected to SCs only, use this parameter to reduce the number of network interfaces for MCs. Example:  

```
mc_num_of_interfaces = 2
```
<table>
<thead>
<tr>
<th>Parameter</th>
<th>Applicable Environment</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sc_instance_type</td>
<td>All</td>
<td>Defines the instance type (virtual machine size / flavor) for SCs. Refer to the <em>Mediant Cloud Edition Installation Manual</em> for a list of officially supported and recommended instance types. Contact AudioCodes support if you want to use a different instance type and to verify that this configuration is allowed and supported. Example: <code>sc_instance_type = Standard_DS3_v2</code></td>
</tr>
<tr>
<td>mc_instance_type</td>
<td>All</td>
<td>Defines the instance type (virtual machine size / flavor) for MCs. Refer to the <em>Mediant Cloud Edition Installation Manual</em> for a list of officially supported and recommended instance types. Contact AudioCodes support if you want to use a different instance type and to verify that this configuration is allowed and supported. Example: <code>mc_instance_type = c4.2xlarge</code></td>
</tr>
</tbody>
</table>
| sc_image_id     | AWS, Azure             | Defines the local image for SCs (instead of the default Marketplace image). Syntax:  
  - AWS: AMI ID  
  - Azure: Resource Group name / image name  
Examples: `sc_image_id = ami-9a50cff5`  
`sc_image_id = rg1/image1` |
| mc_image_id     | AWS, Azure             | Defines the local image for MCs (instead of the default Marketplace image). Syntax:  
  - AWS: AMI ID  
  - Azure: Resource Group name / image name  
Examples: `sc_image_id = ami-9a50cff5`  
`sc_image_id = rg1/image1` |
<p>| sc1_ha_name     | All                    | Defines the name of the first SC on the Web interfaces Monitor page. Example: <code>sc1_ha_name = sc-1</code> |
| sc2_ha_name     | All                    | Defines the name of the second SC on the Web interfaces Monitor page. Example: <code>sc2_ha_name = sc-2</code> |</p>
<table>
<thead>
<tr>
<th>Parameter</th>
<th>Applicable Environment</th>
<th>Description</th>
<th>Syntax</th>
</tr>
</thead>
</table>
| `sc_tags`       | AWS, Azure, Google     | Assigns tags to SCs. | Syntax:  
  - AWS: comma-separated list of name=value pairs  
  - Google: comma-separated list of tags  
  Examples:  
    - `sc_tags = type=sbc,role=sc`
    - `sc_tags = sbc,sc`
| `mc_tags`       | AWS, Azure, Google     | Assigns tags to MCs. | Syntax:  
  - AWS: comma-separated list of name=value pairs  
  - Google: comma-separated list of tags  
  Examples:  
    - `mc_tags = type=sbc,role=mc`
    - `mc_tags = sbc,mc`
| `sc_ini_params` | All                    | Defines additional configuration parameters (in INI file format) for SCs. | Syntax: multiple lines can be specified using `\n` as a line delimiter.  
  Example:  
    - `sc_ini_params = EnableSyslog = 1\nSyslogServerIP = 10.1.2.3`
  **Note:** Use with caution and do not overwrite the default INI configuration parameters created by Stack Manager.
| `mc_ini_params` | All                    | Defines additional configuration parameters (in INI file format) for SCs. | Syntax: multiple lines can be specified using `\n` as a line delimiter.  
  **Note:** Use with caution and do not overwrite the default INI configuration parameters created by Stack Manager.
| `storage_account_type` | Azure       | Defines the storage account type for managed disks. | Valid values include:  
  - Standard_LRS  
  - Premium_LRS  
  - StandardSSD_LRS  
  Example:  
    - `storage_account_type = Premium_LRS`
| `resource_group` | Azure       | Defines the name of the existing Resource Group. | If not empty, stack resources will be deployed into this Resource Group instead of creating a new one. The Resource Group must be empty prior to stack creation.  
  Example:  
    - `resource_group = SbcGroup1`
3.4 Checking Stack State and Configuration

To check the state and configuration of the existing stack, open the Stacks page and then click the specific stack. The Stack Information page is displayed, which allows you to check the current stack state, inspect and modify its configuration, and perform actions such as scale-out, scale-in, and delete the stack if it’s no longer needed.

Figure 3-14: Stack Information Page

3.5 Performing Operations on Stack

You can perform operations on the running stack (e.g., Scale Out), by clicking the corresponding button on the toolbar of the Stack Information page.

All operations, except for Delete and Heal, are serialized and can be performed one at a time. For example, if you started the Scale Out operation, you have to wait until it completes prior to starting the Scale In operation.

The stack state is updated accordingly when an operation is being performed.
3.6 Scaling Mediant CE Stack

The number of active MCs in the Mediant CE stack may vary to match the required service capacity. This is called scaling and ensures that the stack utilizes the optimal amount of resources at any point of time and elastically scales on demand. An operation that increases the amount of active MCs is called *Scale Out*; an operation that decreases the amount of active MCs is called *Scale In*.

To ensure fast and reliable scaling, Stack Manager pre-creates all needed MCs in advance (up to the maximum number) and stops/starts them accordingly during scale in/out operations.

Scaling decision can be triggered either manually—by running the *Scale In*, *Scale Out* or *Scale To* commands—or automatically based on the current cluster utilization.

The size of the cluster is configured by the following two configuration parameters:

- Minimum Number of Media Components
- Maximum Number of Media Components

3.6.1 Scale Out Operation

The *Scale Out* operation increases the number of MCs in the Mediant CE stack, by starting additional pre-created "idle" MCs (for example, corresponding to the AWS EC2 instance state changes from *stopped* to *running*).

You may optionally specify the number of MCs to add to the service.

![Figure 3-15: Scale Out Operation](image)

The *Scale Out* operation is not allowed when *Automatic Scaling* is enabled. Use the *Scale To* operation instead.

3.6.2 Scale In Operation

The *Scale In* operation decreases the number of MCs in the Mediant CE stack, by stopping a certain number of "active" MCs (for example, corresponding to the AWS EC2 instance state changes from *running* to *stopped*).

You may optionally specify the number of MCs to be removed from the service.

![Figure 3-16: Scale In Operation](image)

The *Scale In* operation is not allowed when *Automatic Scaling* is enabled. Use the *Scale To* operation instead.
### 3.6.3 Scale To Operation

The Scale To operation sets the number of MCs in the Mediant CE stack to the specified value. It essentially performs a Scale In or Scale Out operation, depending on the current stack state.

![Figure 3-17: Scale To Operation](image)

In contrast to Scale In and Scale Out operations, the Scale To operation is allowed when Automatic Scaling is enabled. Regardless of whether it adds or removes MCs, for the purposes of calculating a cool down period, the Scale To operation is considered to be equivalent to the Scale Out operation. This means that the cluster size may be increased immediately after completing the Scale To command, if needed.

### 3.7 Automatic Scaling

Automatic Scaling adjusts the Mediant CE cluster size to the current service needs, by measuring current cluster utilization and changing its size accordingly. It is implemented by a background job performed by the Stack Manager.

For every stack that is in "running" state and has Automatic Scaling enabled, Stack Manager calculates the total amount of "free" media and DSP resources, using accumulative percentage points, where 100% corresponds to the capacity of a single MC. For example, for a cluster that is in the following state:

<table>
<thead>
<tr>
<th>id</th>
<th>IP address</th>
<th>status</th>
<th>%media</th>
<th>%dsp</th>
</tr>
</thead>
<tbody>
<tr>
<td>mc-1</td>
<td>172.31.78.116</td>
<td>connected</td>
<td>30</td>
<td>0</td>
</tr>
<tr>
<td>mc-2</td>
<td>172.31.75.42</td>
<td>connected</td>
<td>40</td>
<td>0</td>
</tr>
<tr>
<td>mc-3</td>
<td>172.31.65.5</td>
<td>connected</td>
<td>25</td>
<td>0</td>
</tr>
</tbody>
</table>

Free media resources are calculated as follows:

\[
\text{free_media} = (100-30) + (100-20) + (100-25) = 205\%
\]

**Note:** The calculated number is the number of excessive MCs capacity in the Mediant CE cluster. For example, 100% corresponds to the state where the total amount of excessive capacity equals the capacity of a single MC. In this state, the failure of a single MC has no effect on traffic capacity, thus providing N+1 redundancy for the media cluster.

The calculated number is then compared against Scale In and Scale Out Thresholds, which are defined in the stack configuration. If the number is below the Scale Out Threshold, the Scale Out operation is triggered. If the number is above the Scale In Threshold, the Scale In operation is triggered.
It is possible to disable media or DSP thresholds, by setting them to 0 (zero). If both media and DSP thresholds are used, the decision is made as follows:

- **Scale Out** is performed when *either* media or DSP utilization is below the threshold
- **Scale In** is performed when *both* media and DSP utilization are above the threshold

*Maximum / Minimum Number of Media Components* parameters define the maximum / minimum cluster size, and automatic scaling mechanism takes them into account when making its decisions.

Automatic scaling produces separate logs that you can view through Web or CLI management interfaces:

```
$ tail -f /var/log/stack_mgt/auto_scale.log
300% of media resources in stack 'stack1' are unused
MEDIA_UTIL_SCALE_IN_THRESHOLD is 250
Trigger automatic scale in
Choosing SBC media components to be removed...... done
Preparing SBC media component 'mc-3' for removal.... done
Initializing AWS client... done
Updating SBC cluster configuration.... done
Removing SBC media components............................. done
```

### 3.7.1 Cool Down Period

To prevent stack size 'bouncing', the **Automatic Scaling Cool Down Time** parameter defines the minimum time (in seconds) between consecutive **Scale Out** and **Scale In** decisions.

### 3.7.2 Auto Scale Step

The number of MCs to be added or removed by the automatic scaling mechanism can be configured using the **Automatic Scaling Scale-In / Scale-Out Step** parameters.

Both parameters are set to 1 by default, thus enabling Automatic Scaling to add or remove one MC at a time. If you change the **Automatic Scaling Scale-Out Step** parameter to a greater value (e.g., 2), your stack size will grow quickly to adjust to traffic demands, but will shrink slowly when traffic is reduced.

### 3.7.3 Changing Cluster Size at Specific Time of Day

In certain scenarios, service capacity is typically expected to change at certain times of day. For example, if the Contact Center starts to operate at 9:00 AM, it would be reasonable to expect that SBC traffic will surge at that time.

It is possible to change Mediant CE scaling while having Automatic Scaling enabled, using one of the following methods:

- Changing the **Minimum Number of Media Components** parameter, which defines the minimum cluster size
- Defining the target cluster size by the **Scale To operation**

If you choose to define the target cluster size by the **Scale To** operation, keep in mind that the cool-down period is calculated as if the **Scale Out** operation was performed. Therefore, cluster size will grow immediately if required and will not be reduced for the cool-down period even if traffic hasn’t started yet.

The corresponding operations may be programmed to run at a specified time of day using CLI and the cron scheduler. Make sure that commands are run by the *stack_mgr* user, and...
replace the **stack_mgr** command with the expression "\(\text{"}/usr/bin/python3 /opt/stack_mgr/bin/stack_mgr.py\)". For example:

```bash
$ cat /var/stack_mgr/scale_to.sh
#!/bin/bash
STACK_MGR="/usr/bin/python3 /opt/stack_mgr/bin/stack_mgr.py"
$STACK_MGR scale $1 -n $2 >> /var/log/stack_mgr/cron.log
```

```bash
$ cat /etc/cron.d/stack_mgr
* * * * * stack_mgr /var/stack_mgr/scale_to.sh stack1 10
```

### 3.8 Modifying Stack Configuration

To modify configuration of the existing Mediant VE/CE stack, open the Stack information page, and then click the **Modify** button on the toolbar to open the Modify stack dialog box. Change stack configuration parameters as desired, and then click **Modify** to apply your changes.

**Figure 3-18: Modifying Stack Configuration**

![Modify Stack Configuration](image)

Most of the parameters are applied immediately and have no adverse effect on service. However, some parameters may require an additional **Update** operation.

**Figure 3-19: Modifying Parameter that Requires Update**

![Modify Parameter that Requires Update](image)
3.8.1 Update Operation

The *Update* operation updates the stack to the new configuration. It is required when modified configuration requires applying some changes to the underlying virtual infrastructure resources, for example, when you resize the cluster.

The need to do an *Update* operation is indicated in the *Modify* operation output and on the Stack information page:

*Figure 3-20: Stack in "Update Needed" State*

![Stack Information Page]

Click the **Update** button on the toolbar to start the *Update* operation and wait until it completes.

**Note:** The *Update* operation may be service affecting. It is therefore recommended to run it during a maintenance period.
3.8.2 Modifiable Parameters for Mediant CE

The following table lists all stack configuration parameters that can be modified.

### Table 3-4: Modifiable Stack Configuration Parameters

<table>
<thead>
<tr>
<th>Group Name</th>
<th>Parameter</th>
<th>Applicable Environment</th>
<th>Requires Update</th>
</tr>
</thead>
<tbody>
<tr>
<td>General</td>
<td>Minimum number of media components</td>
<td>All</td>
<td>Yes</td>
</tr>
<tr>
<td></td>
<td>Maximum number of media components</td>
<td>All</td>
<td>Yes</td>
</tr>
<tr>
<td>Automatic scaling</td>
<td>Automatic scaling</td>
<td>All</td>
<td>No</td>
</tr>
<tr>
<td></td>
<td>Media utilization scale in threshold</td>
<td>All</td>
<td>No</td>
</tr>
<tr>
<td></td>
<td>Media utilization scale out threshold</td>
<td>All</td>
<td>No</td>
</tr>
<tr>
<td></td>
<td>DSP utilization scale in threshold</td>
<td>All</td>
<td>No</td>
</tr>
<tr>
<td></td>
<td>DSP utilization scale out threshold</td>
<td>All</td>
<td>No</td>
</tr>
<tr>
<td></td>
<td>Automatic scaling cool down time</td>
<td>All</td>
<td>No</td>
</tr>
<tr>
<td></td>
<td>Automatic scaling scale-in step</td>
<td>All</td>
<td>No</td>
</tr>
<tr>
<td></td>
<td>Automatic scaling scale-out step</td>
<td>All</td>
<td>No</td>
</tr>
<tr>
<td>Signaling Components</td>
<td>Number of network interfaces</td>
<td>AWS</td>
<td>Yes</td>
</tr>
<tr>
<td></td>
<td>Interfaces with public IP</td>
<td>AWS</td>
<td>Yes</td>
</tr>
<tr>
<td></td>
<td>Interfaces with additional IP</td>
<td>AWS</td>
<td>Yes</td>
</tr>
<tr>
<td></td>
<td>Management ports</td>
<td>Azure</td>
<td>Yes</td>
</tr>
<tr>
<td></td>
<td>Signaling ports</td>
<td>Azure</td>
<td>Yes</td>
</tr>
<tr>
<td></td>
<td>Number of network interfaces</td>
<td>AWS</td>
<td>Yes</td>
</tr>
</tbody>
</table>
## 3.8.3 Service Interruption During Stack Update

Upon modification of one of the following configuration parameters, the *Update* operation performs a reboot of SCs and/or MCs:

- **Signaling Components > Number of media interfaces**
- **Media Components > Number of media interfaces**

Reboots are done one at a time, to minimize service downtime. When MCs are rebooted, "inactive" MCs are temporarily brought into service to apply the configuration changes and are shut down afterwards.

## 3.9 Stopping and Starting Stack

If you want to temporarily stop all Mediant CE components (e.g., in a lab environment) use the *Stop* operation. Use the *Start* operation afterwards to return all components back to service.

### Figure 3-22: Stopping Stack

Stop

Do you really want to stop this stack?

Stop  Cancel
3.10 Healing Stack

The Heal operation verifies the state of all stack components and fixes any errors if detected. For example, it can remove MCs that are not properly registered in the SCs or remove orphaned entries from the "Media Components" configuration table.

The command is typically used after Stack Manager is interrupted in the middle of some operation, for example, during stack creation or Scale Out. It can also be useful when the output of some operation (e.g., Scale In) indicates an intermittent failure.

In most cases, Stack Manager heals itself automatically (see the following section). However, in some cases, manual healing is needed to ensure that the stack state matches its configuration.

Figure 3-23: Healing Stack

3.10.1 Automatic Healing

Stack Manager automatically triggers a Heal operation when it detects that an operation (e.g., Scale In or Scale Out) was interrupted. The automatic healing log can be viewed through the Web or CLI management interfaces.
3.11 Deleting Stack

The *Delete* operation deletes the stack and releases all resources allocated during its creation.

![Image of deleting a Stack](image)

3.12 Upgrading Software on Idle Media Components

Upgrading the MCs software is done through the Web interface (**Setup > IP Network > Cluster Manager Settings > Start Upgrade**), as described in the *Mediant Software User’s Manual*. However, this is applicable only to "active" MCs.

To complete upgrade for "idle" MCs (that are in "stopped" state), click the **More > Update Idle MCs** button on the toolbar.

The operation temporarily starts "idle" MCs, waits until they complete software upgrade, and then shuts them down.

3.13 Stack Deployment Details

This section describes the methods that Stack Manager uses to deploy stacks in different virtualization environments. Understanding these details allows you to monitor stack behavior using the virtualization environment’s management interfaces (e.g., AWS dashboard) and to troubleshoot various abnormal scenarios. It is also needed to alter some stack configuration, as described in Section 3.13.2, Adjusting Configuration of Security Groups.

3.13.1 Use of Native Cloud Orchestration

Stack Manager uses native cloud orchestration services to perform stack deployment. This simplifies deployment of multiple stack components and provides tracking for all resources that correspond to the specific stack. Specifically the following services are used:

<table>
<thead>
<tr>
<th>Virtual Environment</th>
<th>Orchestration Service</th>
</tr>
</thead>
<tbody>
<tr>
<td>Amazon Web Services (AWS)</td>
<td>Cloud Formation</td>
</tr>
<tr>
<td>Microsoft Azure</td>
<td>Azure Resource Manager</td>
</tr>
<tr>
<td>Google Cloud</td>
<td>Deployment Manager</td>
</tr>
<tr>
<td>OpenStack</td>
<td>Heat Orchestration Service</td>
</tr>
</tbody>
</table>

For each Mediant CE stack instance, the following orchestration templates are used:

- `<stack_name>-network`: Creates security groups and the cluster interface of SCs
- `<stack_name>-sc`: Creates SC instance(s)
- `<stack_name>-mc-N`: Creates MC instance mc-N (where N is 1, 2, etc.)
3.13.2 Adjusting Configuration of Security Groups

Note: This section is not applicable to Google Cloud environment where Firewall Rules are defined at subnet level and are not managed by Stack Manager.

Stack Manager creates the following Security Groups as part of the <stack_name>-network orchestration template:

- **clusterSecurityGroup**: Defines rules for internal communication between Mediant CE components. It is applied to the interfaces in the Cluster Subnet.
- **oamSecurityGroup**: Defines rules for management (OAM) traffic. It is applied to the interfaces in the Main Subnet.
- **mcSecurityGroup**: Defines rules for signaling and media traffic. It is applied to the interfaces in the Main, Additional 1, and Additional 2 Subnets.

All Security Groups contain a default set of rules needed for normal Mediant CE operation. However, you may need to adjust these rules, for example, to allow signaling traffic on additional interfaces (besides default ports 5060 and 5061) or to limit management traffic to specific machines / subnets.

To adjust Security Group rules:

1. Locate the Security Group in the virtual environment (e.g. AWS VPC) dashboard.
2. Adjust inbound and outbound rules as desired.
3.13.3 Using Pre-Defined Public IP Addresses

Stack Manager assigns Public (Elastic/External/Floating) IP addresses to SCs and MCs based on the `sc_public_ips` and `mc_public_ips` configuration parameters, as described in Section 3.3.5.1, Advanced Configuration for Mediant CE.

By default, it allocates new Public IP addresses and assigns them to the instances. If you want to use pre-defined Public IP addresses instead, you need to add the following parameters to stack’s Advanced Config section:

```
public_ip_<component name>_<interface name> = <ID>
```

where:
- `<component name>` is the name of the component to which you want to assign pre-defined Elastic IP address. Valid component names are "sc", "mc-1", "mc-2", etc.
- `<interface name>` is the name of the network interface to which you want to assign pre-defined Elastic IP addresses. Valid interface names are "eth1", "eth2" and "eth3".
- `<ID>` is the environment-specific Public IP address identifier:
  - AWS: Allocation ID of pre-defined Elastic IP address
  - Azure: Resource Group/Name of pre-defined Public IP address
  - Google and OpenStack: Pre-defined external/floating IP address

For example:

**AWS:**
```
public_ip_sc_eth1 = eipalloc-461b3468
public_ip_mc-1_eth1 = eipalloc-37818019
public_ip_mc-2_eth1 = eipalloc-f51f1edb
```

**Azure:**
```
public_ip_sc_eth1 = Ce1ResourceGroup/ScPublicIP
public_ip_mc-1_eth1 = Ce1ResourceGroup/Mc1PublicIP
```

Stack Manager uses pre-defined Public IP addresses for all user-defined components/interfaces as per the above configuration and allocates new Public IP addresses for all the rest.

3.13.4 Using Pre-Defined Private IP Addresses

Stack Manager assigns Private IP addresses to SCs and MCs based on configured network interfaces and `sc_additional_ips` / `mc_additional_ips` configuration parameters, as described in Section 3.3.5.1, Advanced Configuration for Mediant CE.

By default, IP addresses are dynamically allocated from the corresponding subnets. If you want to specify static private IP addresses instead, you can add the following parameters to the stack configuration file:

```
private_ip_<component name>_<interface name> = <private IPs>
```

where:
- `<component name>` is the name of the component to which you want to assign pre-defined private IP addresses. Valid component names are "sc-1", "sc-2", "mc-1", "mc-2", etc. For Azure, you can also use the "sc" component name to specify a pre-defined private IP address for the Internal Load Balancer.
- `<interface name>` is the name of the network interface to which you want to assign pre-defined private IP addresses. Valid interface names are "eth0", "eth1", "eth2" and "eth3".
- `<private IPs>` is a comma-separated list of private IP addresses. The first address is the primary address while additional addresses are secondary addresses.
The **private_ip_** configuration parameter must specify *all* private IP addresses on the specific network interface of the specific instance. It's impossible to configure some IP addresses of the network interface statically and allocate others dynamically.

Adhere to the following rules when using the **private_ip_** configuration parameter:

### AWS:
- For "sc-1":
  - "eth0" must have two IP addresses.
  - Other interfaces must have two IP addresses plus additional IP addresses, as specified by **sc_additional_ips**.
- For "sc-2":
  - All interfaces must have one IP address.
- For "mc-1", "mc-2" etc.:
  - "eth0" must have one IP address.
  - Other interfaces must have one IP address plus additional IP addresses, as specified by **mc_additional_ips**.

### Azure:
- For "sc-1" and "sc-2":
  - "eth0" must have two IP addresses.
  - Other interfaces must have one IP address plus additional IP addresses, as specified by **sc_additional_ips**.
- For "sc":
  - Applicable only to configurations that use an Internal Load Balancer.
  - Specified IP address is assigned to the Internal Load Balancer interface.
  - One IP address must be specified.
- For "mc-1", "mc-2" etc.:
  - "eth0" must have one IP address.
  - Other interfaces must have one IP address plus additional IP addresses, as specified by **mc_additional_ips**.

### Google Cloud:
- For "sc-1" and "sc-2":
  - "eth0" must have one IP address.
  - "eth1" must have two IP addresses.
- For "mc-1", "mc-2" etc.:
  - "eth1" must have one IP address.
  - Other interfaces must have one IP address plus additional IP addresses, as specified by **mc_additional_ips**.
OpenStack:

- For "sc-1":
  - "eth0" must have two IP addresses.
  - Other interfaces must have one IP address plus additional IP addresses, as specified by `sc_additional_ips`.
- For "sc-2":
  - All interfaces must have one IP address.
- For "mc-1", "mc-2" etc.:
  - "eth0" must have one IP address.
  - Other interfaces must have one IP address plus additional IP addresses, as specified by `mc_additional_ips`.

For example:

```
private_ip_sc-1_eth0 = 172.31.128.1,172.31.129.1
private_ip_sc-1_eth1 = 172.31.68.1,172.31.69.1
private_ip_sc-1_eth2 = 172.31.78.1,172.31.79.1

private_ip_sc-2_eth0 = 172.31.128.2
private_ip_sc-2_eth1 = 172.31.68.2
private_ip_sc-2_eth2 = 172.31.78.2

private_ip_mc-1_eth0 = 172.31.128.101
private_ip_mc-1_eth1 = 172.31.68.101
private_ip_mc-1_eth2 = 172.31.78.101

private_ip_mc-2_eth0 = 172.31.128.102
private_ip_mc-2_eth1 = 172.31.68.102
private_ip_mc-2_eth2 = 172.31.78.102
```
4 CLI Interface

4.1 Accessing CLI Interface

Stack Manager's CLI is accessed by switching to the stack_mgr user, using the following command:

```
$ stack_mgr_cli
```

If the above command doesn't work, use the following alternative command to do the same:

```
$ sudo su - stack_mgr
```

4.2 Invocation

Most of the Stack Manager CLI is provided using the stack_mgr command. Auto-completion is available for sub-commands and optional parameters.

4.3 Usage Information

Brief usage information is provided by running the stack_mgr command without arguments:

```
$ stack_mgr
```

usage: stack_mgr [-h] [--version]
{create,delete, purge, list, show, scale-out, scale-in, heal, auto-scale, update}
...

More detailed usage information is provided when '-h' or '--help' arguments are specified:

```
$ stack_mgr --help
```

usage: stack_mgr [-h] [--version]
{create, delete, purge, list, show, scale-out, scale-in, heal, auto-scale, update}
...

AudioCodes Stack Manager

positional arguments:
{create, delete, purge, list, show, scale-out, scale-in, heal, auto-scale, update, modify, reboot, stop, start, configure}

create                  create stack
delete                  delete stack
list                    list stacks
show                    show stack
scale-out               scale out stack
scale-in                scale in stack
scale                   scale stack
heal                    heal stack
modify                  modify stack configuration
update                  update stack
stop                    stop stack
4.4 Global Configuration

The **configure** command performs Stack Manager configuration:

```
$ stack_mgr configure --help
```

Usage:
```
usage: stack_mgr configure [-h] [--aws-access-key ACCESS_KEY]
[--aws-secret-key SECRET_KEY]
[--aws-s3-bucket BUCKET]
[--aws-verify]
[--name-prefix PREFIX]
[--rest-api-username USERNAME]
[--rest-api-password PASSWORD]
[--debug-log {enable,disable}]
[--azure-tenant-id ID]
[--azure-client-id ID]
[--azure-secret AZURE_SECRET]
[--azure-subscription-id ID]
[--azure-blob-account-name NAME]
[--azure-blob-account-key KEY]
[--azure-blob-container CONTAINER]
[--azure-verify]
[--openstack-cloud-name NAME]
[--openstack-container CONTAINER]
[--openstack-verify]
[--google-credentials CREDENTIALS]
[--google-project PROJECT]
[--google-storage-bucket BUCKET]
[--google-storage-prefix PREFIX]
[--google-verify]
```

Optional arguments:
- `-h, --help` show this help message and exit
- `--version` show program's version number and exit
4. CLI Interface

REST API password
--debug-log {enable,disable}  
  debug log
--azure-tenant-id AZURE_TENANT_ID  
  Azure tenant id
--azure-client-id AZURE_CLIENT_ID  
  Azure client id
--azure-secret AZURE_SECRET  
  Azure secret
--azure-subscription-id AZURE_SUBSCRIPTION_ID  
  Azure subscription id
--azure-blob-account-name AZURE_BLOB_ACCOUNT_NAME  
  Azure blob account name
--azure-blob-account-key AZURE_BLOB_ACCOUNT_KEY  
  Azure blob account key
--azure-blob-container AZURE_BLOB_CONTAINER  
  Azure blob container
--azure-verify  
  Verify access to Azure API
--openstack-cloud-name OPENSTACK_CLOUD_NAME  
  Openstack cloud name
--openstack-container OPENSTACK_CONTAINER  
  Openstack container name
--openstack-verify  
  Verify access to OpenStack API
--google-credentials GOOGLE_CREDENTIALS  
  Google application credentials
--google-project GOOGLE_PROJECT  
  Google project name
--google-storage-bucket GOOGLE_STORAGE_BUCKET  
  Google storage bucket name
--google-storage-prefix GOOGLE_STORAGE_PREFIX  
  Google storage file prefix
--google-verify  
  Verify access to Google API

To show current configuration, use the command without any arguments.
To update a specific configuration parameter(s), use the command with arguments.

4.5 Listing Available Stacks

The **list** command lists available stacks.

```
$ stack_mgr list
+--------+-------------+-----+-------------+
| name   | type        | vim | state       |
+--------+-------------+-----+-------------+
| stack1 | sbc-cluster | aws | idle        |
| stack2 | sbc-cluster | aws | scaling-out |
+--------+-------------+-----+-------------+
```
4.6 Creating a New Stack

Creation of a new stack through CLI consists of the following steps:

1. Creating the stack configuration file, which can be done using one of the following methods:
   - SBC Cluster Configuration Tool (recommended) – see Section 4.6.1, Creating Stack Configuration File via SBC Cluster Configuration Tool (Recommended Method) for more information
   - Manually, by editing provided reference files – see Section 4.6.2, Creating Stack Configuration File Manually (Alternative Method) for more information
2. Creating the stack by the `create` command.

The stack configuration file contains configuration parameters of the created stack. The same configuration file can be used to create multiple stacks.

4.6.1 Creating Stack Configuration File via SBC Cluster Configuration Tool (Recommended Method)

The SBC Cluster Configuration Tool provides a simple interactive user interface (UI) for creating the configuration file.

- To create the stack configuration file for Mediant CE, type `sbc_cluster_config`
- To create the stack configuration file for Mediant VE, type `sbc_config`

You are prompted for the basic Mediant VE/CE configuration parameters, and a new configuration file will be created. You may use this file to create the Mediant VE/CE instance using the `stack_mgr create` command, as described in Section 4.6.3, Creating a New Stack. It is recommended to review the created file prior to the instance creation and modify it if needed.

**Note:** The following output is provided as an example only and therefore, may not be up to date.

```
$ sbc_cluster_config
----------------------------------
SBC Cluster Configuration Tool
----------------------------------

This tool creates a configuration file that may be used to create the Mediant CE cluster using the `stack_mgr create` command.

Enter configuration file name: `stack1.cfg`

List of AWS regions:
+------------------+-------+
| #    | name  |
+------------------+-------+
```
Choose region: **11**

List of AWS VPCs:

<table>
<thead>
<tr>
<th>#</th>
<th>id</th>
<th>name</th>
<th>cidr block</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>vpc-45f3152c</td>
<td>MAIN_VPC</td>
<td>172.31.0.0/16</td>
</tr>
<tr>
<td>2</td>
<td>vpc-3440505c</td>
<td>QAAUTO_VPC</td>
<td>10.10.0.0/16</td>
</tr>
<tr>
<td>3</td>
<td>vpc-39d23352</td>
<td>QAOVOC</td>
<td>172.16.138.0/24</td>
</tr>
</tbody>
</table>

Choose VPC: **1**

Copy the official Mediant VE SBC AMI published by AudioCodes, to your account and use it instead of the public one. This ensures that you will be able to resize the cluster even if public AMI becomes unavailable, for some reason.

Enter AMI ID: **ami-012ec16ee68bd011d**

Key Pair is used to provide secure access to the SBC cluster's CLI interface using the SSH protocol. It is mandatory for the AWS environment even though the SBC, in its default configuration, supports the SSH login using username/password.

<table>
<thead>
<tr>
<th>#</th>
<th>name</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>qa_key</td>
</tr>
<tr>
<td>2</td>
<td>qa_auto_key</td>
</tr>
<tr>
<td>3</td>
<td>aws_ssh_frankfurt_1</td>
</tr>
</tbody>
</table>
Choose key pair: **3**

You must create the IAM role that allows the SBC to manage its IP addresses. The role must look like, as follows:
```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Action": [
        "ec2:AssignPrivateIpAddresses",
        "ec2:UnassignPrivateIpAddresses",
        "ec2:AssociateAddress",
        "ec2:DescribeAddresses",
        "ec2:DescribeNetworkInterfaceAttribute",
        "ec2:DescribeNetworkInterfaces"
      ],
      "Effect": "Allow",
      "Resource": "*"
    }
  ]
}
```

Refer to the Mediant CE Installation Manual for more information.

Enter IAM role: **SBC-HA-3**

Instance type of Signaling Components (SC) is r4.2xlarge.
Instance type of Media Components (MC) depends on their profile.

<table>
<thead>
<tr>
<th>#</th>
<th>mc profile</th>
<th>instance type</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>forwarding</td>
<td>r4.large</td>
</tr>
<tr>
<td>2</td>
<td>transcoding</td>
<td>c4.4xlarge</td>
</tr>
</tbody>
</table>

Enter media components profile: **1**

Signaling components (SC) may have 2, 3 or 4 network interfaces that are connected as follows:

<table>
<thead>
<tr>
<th>iface</th>
<th>subnet_id</th>
<th>traffic</th>
</tr>
</thead>
<tbody>
<tr>
<td>eth0</td>
<td>cluster</td>
<td>internal cluster communication</td>
</tr>
<tr>
<td>eth1</td>
<td>oam</td>
<td>management (HTTP, SSH) + signaling (SIP)</td>
</tr>
<tr>
<td>eth2</td>
<td>signaling1</td>
<td>signaling (SIP)</td>
</tr>
<tr>
<td>eth3</td>
<td>signaling2</td>
<td>signaling (SIP)</td>
</tr>
</tbody>
</table>
+-------+------------+------------------------------------------+

Notes:
- Primary IP addresses are not used except for "eth0" (cluster interface).
- Secondary IP addresses are used instead and "float" across the two SC instances (in HA configuration).

Enter the number of interfaces for signaling components (2, 3 or 4): 2

To communicate with signaling components from outside the AWS cloud, Elastic IP addresses must be assigned to the relevant network interfaces.

Provide a comma-separated list of SC network interfaces that will be assigned with Elastic IP addresses.
For example: "eth1" or "eth1,eth2"

Notes:
- In a typical setup, Elastic IP is assigned to "eth1"
- Elastic IP cannot be assigned to "eth0" because it is used for internal cluster communication
- If all management and signaling communication occurs inside the AWS cloud and therefore, you do not need Elastic IPs, press Enter to continue

Enter value: eth1

Media components (MC) may have 2, 3 or 4 network interfaces that are connected as follows:

<table>
<thead>
<tr>
<th>iface</th>
<th>subnet_id</th>
<th>traffic</th>
</tr>
</thead>
<tbody>
<tr>
<td>eth0</td>
<td>cluster</td>
<td>internal cluster communication</td>
</tr>
<tr>
<td>eth1</td>
<td>oam</td>
<td>management (HTTP, SSH) + media (RTP)</td>
</tr>
<tr>
<td>eth2</td>
<td>signaling1</td>
<td>media (RTP)</td>
</tr>
<tr>
<td>eth3</td>
<td>signaling2</td>
<td>media (RTP)</td>
</tr>
</tbody>
</table>
+-------+------------+--------------------------------------+

Notes:
- Primary IP addresses are available on all interfaces.

Enter the number of interfaces for media components (2, 3 or 4): 2
To communicate with media components from outside the AWS cloud, Elastic IP addresses must be assigned to the relevant network interfaces.

Provide comma-separated list of MC network interfaces that will be assigned with Elastic IP addresses, For example: "eth1" or "eth1,eth2"

Notes:
- In a typical setup, Elastic IP is assigned to "eth1"
- Elastic IP cannot be assigned to "eth0" because it is used for internal cluster communication
- If all media communication occurs inside the AWS cloud and therefore, you do not need Elastic IPs, press Enter to continue

Enter value: eth1

Cluster subnet carries internal traffic between SBC cluster components and is used for accessing the AWS API. It must have the NAT Gateway attached, as all communication (including AWS API access) is done using private IP addresses. The subnet is connected to both signaling (SC) and media (MC) components. Use a dedicated subnet and protect it from unauthorized access.

<table>
<thead>
<tr>
<th>#</th>
<th>id</th>
<th>name</th>
<th>cidr range</th>
<th>avail. zone</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>subnet-be6e8bc3</td>
<td>cluster</td>
<td>172.31.48.0/20</td>
<td>eu-central-1b</td>
</tr>
<tr>
<td>2</td>
<td>subnet-e19c649c</td>
<td>test</td>
<td>172.31.64.0/20</td>
<td>eu-central-1a</td>
</tr>
<tr>
<td>3</td>
<td>subnet-1536d368</td>
<td>oam</td>
<td>172.31.0.0/2</td>
<td>eu-central-1b</td>
</tr>
</tbody>
</table>

Cluster subnet: 1

OAM subnet carries management (HTTP, SSH, etc.), signaling (SIP) and media (RTP) traffic. It is connected to both signaling (SC) and media (MC) components.

<table>
<thead>
<tr>
<th>#</th>
<th>id</th>
<th>name</th>
<th>cidr range</th>
<th>avail. zone</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>subnet-1536d368</td>
<td>oam</td>
<td>172.31.0.0/2</td>
<td>eu-central-1b</td>
</tr>
</tbody>
</table>

OAM subnet: 1
The size of the cluster, and specifically the number of media components, may vary to match the required service capacity. This ensures that the cluster utilizes optimal amount of resources at any point of time and elastically scales on demand.

The scaling decision may be done either manually - by executing 'scale-in' or 'scale-out' commands - or automatically based on the current cluster utilization.

The size of the cluster is controlled by the following two parameters:
* Minimum Number of Media Components
* Maximum Number of Media Components

To ensure fast scaling, Stack Manager pre-creates all needed media components in advance (up to the maximum number) and stops/starts them accordingly during scale in/out operations.

Minimum Number of Media Components (0-21): 2
Maximum Number of Media Components (2-21): 5

Credentials for management interface.

Username: **sbcadmin**
Password:
Retype password:

Creating configuration file stack1.cfg
Done

**Note:** When selecting the region, VPC, subnets and other listed objects, enter either a corresponding row number (e.g., "1") or an Object ID (e.g., "vpc-45f3152c").
4.6.2 Creating Stack Configuration File Manually (Alternative Method)

As an alternative to running the SBC Cluster Configuration Tool (described in the previous section), you can create the stack configuration file manually by copying it from the /opt/stack_mgr.cfg directory and then modifying it using a text editor tool.

You can edit the copied file in one of the following ways:

- On the server itself, by using, for example, a "vi" or "nano" editor:
  
  ```bash
  $ cp /opt/stack_mgr/cfg/sbc-cluster-aws.cfg stack1.cfg
  $ vi stack1.cfg
  ```

- By transferring the copied file from the server through SFTP/SCP to a computer, modifying it using a standard text editor (e.g., Notepad), and then transferring it back to the server.

When you create the stack configuration file manually, make sure that the following parameters are updated:

- **Amazon Web Services (AWS):**
  - `aws_region`: Defines the AWS region where the Mediant CE stack will be deployed.
  - `vpc_id`: Defines the VPC where the Mediant CE stack will be deployed.
  - `*subnet_id`: Defines the subnet IDs for all applicable subnets.
  - `ssh_key_pair`: Defines the SSH key pair for connecting to the Mediant CE CLI.
  - `*image_id`: Defines the AMI ID of the local copy of the Mediant VE/CE image.
  - `sc_iam_role`: Defines the SBC IAM Role name. Refer to the Mediant Cloud Edition Installation Manual for detailed instructions on how to create this role.

- **Microsoft Azure:**
  - `location`: Defines the Azure location where the Mediant CE stack will be deployed.
  - `vnet_id`: Defines the Virtual Network where the Mediant CE stack will be deployed.
  - `*subnet_id`: Defines the subnet name for all applicable subnets.

- **Google Cloud:**
  - `region`: Defines the Google Cloud region where the Mediant CE stack will be deployed.
  - `*subnet_id`: Defines the subnet name for all applicable subnets.
  - `*image_id`: Defines the Image ID of the Mediant VE/CE image.

- **OpenStack:**
  - `*subnet_id`: Defines the subnet name for all applicable subnets.
  - `*image_id`: Defines the image name of the Mediant VE/CE image.
  - `*instance_type`: Defines the flavor of the Mediant CE instances.
4.6.2.1 Sample Configuration File

The following is a sample configuration file for Mediant CE in the AWS cloud:

```
# -----------------
# Stack descriptor
# -----------------

# stack type
stack_type = sbc-cluster

# virtual infrastructure manager
vim = aws

# ------------------
# Generic parameters
# ------------------

# Initial cluster size
mc_num = 2

# Minimal cluster size
min_mc_num = 2

# Maximum cluster size
max_mc_num = 5

# --------------------------
# Auto-scaling configuration
# --------------------------

# Auto-scaling - enable/disable
auto_scale = disable

# Media utilization scale in threshold - in accumulative free
# percentage points (when auto-scaling is enabled and total
# amount of free resources in the cluster raises above this
# threshold, scale-in is triggered)
media_util_scale_in_threshold = 250

# Media utilization scale out threshold - in accumulative free
# percentage points (when auto-scaling is enabled and total
# amount of free resources in the cluster falls below this
```

Note: The file is provided as an example only and therefore, may not be up to date. Use files from the `/opt/stack_mgr/cfg` directory when creating a new stack configuration file.
# threshold, scale-in is triggered)
media_util_scale_out_threshold = 100

# DSP utilization scale in threshold - in accumulative free
# percentage points (when auto-scaling is enabled and total
# amount of free resources in the cluster raises above this
# threshold, scale-in is triggered)
dsp_util_scale_in_threshold = 0

# DSP utilization scale out threshold - in accumulative free
# percentage points (when auto-scaling is enabled and total
# amount of free resources in the cluster falls below this
# threshold, scale-in is triggered)
dsp_util_scale_out_threshold = 0

# Auto-scaling cool down time in seconds
# (minimum time between two consecutive 'opposite' auto-scaling
# operations, e.g., scale-out after scale-in)
auto_scale_cooldown_time = 900

# Auto-scaling scale-in step
# (number of media instances to be removed)
auto_scale_in_step = 1

# Auto-scaling scale-out step
# (number of media instances to be added)
auto_scale_out_step = 1

# ---------------------
# Network configuration
# ---------------------

# AWS region name
# (use 'aws ec2 describe-regions' command to find all
# available regions)
aws_region = eu-central-1

# VPC where stack is deployed
vpc_id = vpc-45f3152c

# SBC cluster requires the following subnets:
#   - cluster   - for internal communication between
#                  cluster nodes
#   - oam       - for management (HTTP, SSH),
#                  signaling (SIP) and media (RTP) traffic
#   - signaling - for signaling (SIP) traffic
#   - media     - for media (RTP) traffic
#
# Note:
#   - During normal cluster operation, only active Signaling
# Component (SC) is accessed for management purposes
# (Web / CLI / SNMP / REST)
#
# It is acceptable to specify the same value for all below 
# subnet_IDs, except for cluster_subnet_id.

cluster_subnet_id = subnet-be6e8bc3
oam_subnet_id = subnet-1536d368
signaling1_subnet_id =
signaling2_subnet_id =
media1_subnet_id =
media2_subnet_id =

# Key Pair provides secure access to the SBC cluster's
# CLI interface via SSH protocol. It is mandatory for the AWS
# environment even though SBC in its default configuration
# supports SSH login using username/password.
ssh_key_pair = aws_ssh_frankfurt_1

# --------------------------------------
# Signaling Component (SC) configuration
# --------------------------------------

# 1+1 HA mode - enable / disable
sc_ha_mode = enable

# Signaling Components (SC) network interfaces are connected
# as follows:
# - eth0: cluster
# - eth1: oam
# - eth2: signaling1
# - eth3: signaling2
#
# At least two network interfaces are required.
# Notes:
# - Primary IP addresses are not used except for "eth0" (cluster 
#   interface). Secondary IP addresses are used instead and
#   'float' across the two SC instances (in HA configuration).

# Number of network interfaces - valid values: 2, 3, 4
sc_num_of_interfaces = 2

# Comma-separated list of network interfaces will be assigned
# with Public IP addresses (Elastic IPs).
# For example: "eth1,eth3"
# Notes:
# - In a typical setup, Public IP is assigned to "eth1"
# - Public IP cannot be assigned to "eth0" because it's used
#   for internal cluster communication
# - if all management and signaling communication occurs inside
sc_public_ips = eth1

sc_additional_ips =

sc_instance_type = r4.2xlarge

sc_image_id = ami-9a50c8f5

sc_disk_size = 100

sc_iam_role = SBC-HA-3

sc_ini_file_url =

sc_ini_file_contains_admin_user = true

# Media Component (MC) configuration

# Media Components (MC) network interfaces are connected as follows:
#   - eth0: cluster
#   - eth1: oam
#   - eth2: media1
#   - eth3: media2
#
# At least two network interfaces are required.
# Primary IP addresses are available on all interfaces.
# Number of network interfaces - valid values: 2, 3, 4
mc_num_of_interfaces = 2

# Comma-separated list of network interfaces that will be assigned
# with Public IP addresses (Elastic IPs)
# For example: "eth1,eth3"
# Notes:
# - In a typical setup, Public IP is assigned to "eth1"
# - Public IP cannot be assigned to "eth0" because it's used
#   for internal cluster communication
# - if all media communication happens inside the Azure cloud
#   and therefore, you do not need Public IPs, leave this
#   field blank
mc_public_ips = eth1

# Comma-separated list of network interface names and number of
# corresponding additional IP addresses
# For example: "eth1:2" means "two additional IP addresses
# on eth1"
mc_additional_ips =

# AWS instance type
# Recommended types are:
# - r4.large for media forwarding
# - c4.4xlarge for transcoding
mc_instance_type = r4.large

# AWS image id
mc_image_id = ami-9a50cff5

# Media component profile - forwarding / transcoding
mc_profile = forwarding

# Media component max rate limit (in kpps)
# In addition to numeric values the following special string
# values are supported:
# - "auto" means that PPS limit is automatically calculated
#   based on instance type
# - "unlimited" means that no limit is imposed
mc_max_pps_limit = auto

# ------------------------
# Additional configuration
# ------------------------

# Prefix to be added to all created components
# (note that there is also global stack_mgr configuration
# parameter with a similar name, but this one overrides it if
# set to non-empty value)
name_prefix =
Manage SBC cluster via HTTPS or HTTP protocol - valid values:

* enable / disable

*(change this to Disable if, for example, your firewall intercepts HTTPS connections and blocks them due to self-signed certificate being used)*

manage_via_https = enable

Sample configuration files for additional environments are available in the `/opt/stack_mgr/cfg` directory.

### 4.6.3 Creating a New Stack

After creating the stack configuration file, use the `create` command to create a new stack. Specify the stack name and provide the stack configuration file.

```bash
$ stack_mgr create --help
```

```
usage: stack_mgr create [-h] name cfg_file

positional arguments:
  name                  Name of the stack; may contain letters, numbers and dash symbol only (spaces are not allowed)
  cfg_file              configuration file

optional arguments:
  -h, --help            Show this help message and exit
```

**Note:** Prior to creating a Mediant CE stack instance(s), make sure that all pre-requisites specified in the *Mediant Cloud Edition Installation Manual* are met. The document can be downloaded from AudioCodes website at [https://www.audiocodes.com/library/technical-documents](https://www.audiocodes.com/library/technical-documents).

The `create` process takes a few minutes and detailed progress information is displayed on the console:

```bash
$ stack_mgr create stack1 sbc-cluster.cfg
Initializing AWS client... done
Creating SBC network resources.......................... done
Creating SBC media components............................ done
Creating SBC signaling components..................... done
Waiting until signaling components are ready......... done
Waiting until media components are ready.... done
Removing media components 'mc-3, mc-4, mc-5' from SBC configuration
Removing media components 'mc-3, mc-4, mc-5'... done
Stopping components 'mc-3, mc-4, mc-5'................. done
Use http://52.58.15.164 to connect to the management interface. Stack 'stack1' is successfully created
```

After the `create` command completes, you can connect to the Mediant CE’s management interface through Web or SSH. The corresponding URL is shown in the summary following the stack creation.
Use the credentials provided in the stack configuration file to log in to the Mediant CE management interface.

4.7 Checking Stack State and Configuration

The `show` command displays detailed information about a specific stack. You must specify a valid stack name.

```
$ stack_mgr show --help
usage: stack_mgr show [-h] name

positional arguments:
  name        name of the stack

optional arguments:
  -h, --help  show this help message and exit
  --no-status do not show real-time status
  --idle-mcs  show 'idle' media components
```

```
$ stack_mgr show stack1

Name          : stack1
Type          : sbc-cluster
VIM           : aws
State         : idle
Created at    : May 09, 2018 08:55:29
Region        : eu-central-1
VPC           : vpc-45f3152c

--------------------
Signaling Components
--------------------
Instance type : r4.2xlarge
Image ID      : ami-d771563c
Disk size     : 100 GB

+-----------------+-----------------+-----------------+--------+
| id   | IP address    | status | type   |
+-----------------+-----------------+-----------------+--------+
| sc-1 | 172.31.71.211 | running | r4.2xlarge |
| sc-2 |                  | running | r4.2xlarge |
+-----------------+-----------------+-----------------+--------+

Network configuration:
+--------------------+-----------------+-----------------+--------+
| interface | subnet | id              | status |
+--------------------+-----------------+-----------------+--------+
| eth0     | cluster | subnet-be6e8bc3 | in-use |
```
|     |     | subnet-1536d368 | in-use |
| eth1 | oam |              |        |
| eth2 | signaling1 |       |        |
| eth3 | signaling2 |       |        |

+-------------------------------+
SC number of network interfaces : 2
SC interfaces with public IPs : eth1
SC interfaces with additional IPs :

Media Components

Instance type : r4.large
Image ID : ami-d771563c
Profile : forwarding
Max rate limit : auto

+---------------------------------+
<table>
<thead>
<tr>
<th>id</th>
<th>IP address</th>
<th>status</th>
<th>%media</th>
<th>%dsp</th>
<th>type</th>
</tr>
</thead>
<tbody>
<tr>
<td>mc-1</td>
<td>172.31.67.240</td>
<td>connected</td>
<td>0</td>
<td>-</td>
<td>r4.large</td>
</tr>
<tr>
<td>mc-2</td>
<td>172.31.67.15</td>
<td>connected</td>
<td>0</td>
<td>-</td>
<td>r4.large</td>
</tr>
</tbody>
</table>
+---------------------------------+
Number of media components : 2
Connected media components : 2
Free media resources : 200%
Free DSP resources : -

Network configuration:

+---------------------------------+
<table>
<thead>
<tr>
<th>interface</th>
<th>subnet</th>
<th>id</th>
<th>status</th>
</tr>
</thead>
<tbody>
<tr>
<td>eth0</td>
<td>cluster</td>
<td>subnet-be6e8bc3</td>
<td>in-use</td>
</tr>
<tr>
<td>eth1</td>
<td>oam</td>
<td>subnet-1536d368</td>
<td>in-use</td>
</tr>
<tr>
<td>eth2</td>
<td>media1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>eth3</td>
<td>media2</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
+---------------------------------+
MC number of network interfaces : 2
MC interfaces with public IPs : eth1
MC interfaces with additional IPs :

Min number of media components : 2
Max number of media components : 10
Automatic scaling : enable
Media utilization scale in threshold : 250%
Media utilization scale out threshold : 100%
DSP utilization scale in threshold : 0 (disabled)
 Unless the `--no-status` argument is specified, Stack Manager collects the following additional information:

- **For SCs:**
  - Runtime status (running/stopped), using the cloud-specific API
  - Active instance that currently holds the "public IP", using the cloud-specific API

- **For MCs:**
  - Runtime status (running/stopped), using the cloud-specific API
  - Connectivity status (connected/disconnected), using the SBC REST API
  - Media and DSP utilization, using the SBC REST API

If the `--no-status` argument is specified or the Stack Manager fails to communicate with the SBC cluster, it displays an internal state of the component instead.

### 4.7.1 Checking Idle Media Components

The number and detailed status of MCs reported by the `show` command corresponds to the "active" (running) MCs. "Inactive" (stopped) MCs can be viewed by adding the `--idle-mcs` argument to the `show` command, or by using the virtual environment's (e.g., AWS EC2) dashboard – corresponding instances are in the "stopped" state.

```
$ stack_mgr show stack1 --idle-mcs
...
----------------
Media Components
----------------
Instance type : r4.large
Image ID      : ami-d771563c
Profile       : forwarding
Max rate limit: auto
+-----------------+-----------------+--------+--------+--------+-----------------+
| id   | IP address    | status | %media | %dsp  | type     |
+-----------------+-----------------+--------+--------+--------+-----------------+
| mc-1 | 172.31.67.240 | connected | 0      | -     | r4.large |
| mc-2 | 172.31.67.15  | connected | 0      | -     | r4.large |
| mc-3 | 172.31.70.66  | down    | -      | -     | r4.large |
| mc-4 | 172.31.75.108 | down    | -      | -     | r4.large |
| mc-5 | 172.31.67.179 | down    | -      | -     | r4.large |
+-----------------+-----------------+--------+--------+--------+-----------------+
```
4.8 Scaling Mediant CE Stack

The number of active MCs in the Mediant CE stack may vary to match the required service capacity. This is called scaling and ensures that the stack utilizes the optimal number of resources at any point of time and elastically scales on demand. Operation that increases the number of active MCs is called Scale Out. Operation that decreases the number of active MCs is called Scale In.

4.8.1 Scale Out Operation

The Scale Out operation increases the number of MCs in the Mediant CE stack by starting additional pre-created "idle" MCs (for example, corresponding to the AWS EC2 instance state changes from stopped to running).

You must specify a valid stack name and may optionally specify a number of MCs to be added to the service. If the number of MCs is not specified, one MC is added.

$ stack_mgr scale-out --help
usage: stack_mgr scale-out [-h] [-n num] name

positional arguments:
  name               of the stack

optional arguments:
  -h, --help         show this help message and exit
  -n num, --num  number of media components to be added

The scale-out command is not allowed when Automatic Scaling is enabled. Use the scale command instead.

$ stack_mgr scale-out stack1
The following media components will be brought into service: mc-3
Checking that configuration is allowed... done
Initializing AWS client... done
Starting components 'mc-3'............. done
Successfully started 'mc-3'
Adding media components 'mc-3' to SBC configuration... done
Waiting until media components are ready...... done

4.8.2 Scale In Operation

The Scale In operation decreases the number of MCs in the Mediant CE stack by stopping a certain number of "active" MCs (for example, corresponding to the AWS EC2 instance state changes from running to stopped).

You must specify the valid stack name and may optionally specify one of the following:

- Number of MCs to be removed from the service
- Names of specific MCs to be removed from the service
If none of the above parameters are specified, one MC is removed. If you do not specify MC names, Stack Manager automatically removes MCs with the lowest media utilization:

$ stack_mgr scale-in --help
usage: stack_mgr scale-in [-h] [-n num] [-i ids] name

positional arguments:
  name               name of the stack

optional arguments:
  -h, --help         show this help message and exit
  -n num, --num num  number of media components to be removed
  -i ids, --ids ids  comma-separated list of media component ids to be removed, e.g. mc-3,mc-4

The `scale-in` command is not allowed when Automatic Scaling is enabled. Use `scale` command instead.

$ stack_mgr scale-in stack1
Choosing media components to be taken out of service...... done
The following media components will be taken out of service: mc-3
Checking that configuration is allowed... done

Initializing AWS client... done
Removing media components 'mc-3' from SBC configuration
Locking media component 'mc-3'.... done
Removing media components 'mc-3'... done
Stopping components 'mc-3'................... done

4.8.3 Scale To Operation

`Scale To` operation sets the number of MCs in the Mediant CE stack to the specified value. It essentially performs `Scale In` or `Scale Out` operation, depending on the current stack state. You must specify the valid stack name and a number of active MCs in the cluster.

$ stack_mgr scale --help
usage: stack_mgr scale [-h] [-n num] name

positional arguments:
  name               name of the stack

optional arguments:
  -h, --help         show this help message and exit
  -n num, --num num  number of media components

Contrary to `Scale In` and `Scale Out` operations, the `Scale To` operation is allowed when Automatic Scaling is enabled. Regardless of whether it adds or removes MCs, for the purposes of calculating a cool down period, the `Scale To` operation is considered to be equivalent to the `Scale Out` operation. This means that cluster size may be increased immediately after completing the `Scale To` command, if needed.

$ stack_mgr scale stack1 -n 3
The following media components will be brought into service: mc-3
Checking that configuration is allowed... done
Initializing AWS client... done
Starting components 'mc-3'.......... done
Successfully started 'mc-3'
Adding media components 'mc-3' to SBC configuration... done
Waiting until media components are ready...... done

4.9 Modifying Stack Configuration

The modify command modifies the configuration of the stack.

$ stack_mgr modify --help
usage: stack_mgr modify [-h] [--max-mc-num MAX_MC_NUM]
                  [--min-mc-num MIN_MC_NUM]
                  [--auto-scale {enable,disable}]
                  [--media-util-scale-in-threshold VALUE]
                  [--media-util-scale-out-threshold VALUE]
                  [--dsp-util-scale-in-threshold VALUE]
                  [--dsp-util-scale-out-threshold VALUE]
                  [--auto-scale-cooldown-time TIME]
                  [--auto-scale-in-step {1,2,3,4,5}]
                  [--auto-scale-out-step {1,2,3,4,5}]
                  [--manage-via-https {enable,disable}]
                  [--sc-num-of-interfaces {2,3,4}]
                  [--sc-public-ips SC_PUBLIC_IPS]
                  [--sc-additional-ips SC_ADDITIONAL_IPS]
                  [--mc-num-of-interfaces {2,3,4}]
                  [--mc-public-ips MC_PUBLIC_IPS]
                  [--mc-additional-ips MC_ADDITIONAL_IPS]
                  [--signaling1-subnet-id SUBNET_ID]
                  [--signaling2-subnet-id SUBNET_ID]
                  [--medial-subnet-id SUBNET_ID]
                  [--media2-subnet-id SUBNET_ID]
                  [--sc-instance-type SC_INSTANCE_TYPE]
                  [--mc-instance-type MC_INSTANCE_TYPE]
                  [--sc-image-id SC_IMAGE_ID]
                  [--mc-image-id MC_IMAGE_ID]

positional arguments:
  name                  name of the stack

optional arguments:
  -h, --help            show this help message and exit
  --max-mc-num MAX_MC_NUM
                        maximum number of media components
  --min-mc-num MIN_MC_NUM
                        minimum number of media components
  --auto-scale {enable,disable}
auto scaling

--media-util-scale-in-threshold MEDIA_UTIL_SCALE_IN_THRESHOLD
  media utilization scale in threshold
  (in accumulative free percentage points)

--media-util-scale-out-threshold MEDIA_UTIL_SCALE_OUT_THRESHOLD
  media utilization scale out threshold
  (in accumulative free percentage points)

--dsp-util-scale-in-threshold DSP_UTIL_SCALE_IN_THRESHOLD
  dsp utilization scale in threshold
  (in accumulative free percentage points)

--dsp-util-scale-out-threshold DSP_UTIL_SCALE_OUT_THRESHOLD
  dsp utilization scale out threshold
  (in accumulative free percentage points)

--auto-scale-cooldown-time AUTO_SCALE_COOLDOWN_TIME
  auto scaling cooldown time (in seconds)

--auto-scale-in-step \{1,2,3,4,5\}
  auto scaling scale-in step

--auto-scale-out-step \{1,2,3,4,5\}
  auto scaling scale-out step

--manage-via-https \{enable,disable\}
  use HTTPS or HTTP protocol for
  cluster management

--sc-num-of-interfaces \{2,3,4\}
  number of interfaces for signaling
  components

--sc-public-ips SC_PUBLIC_IPS
  SC interfaces that will be assigned
  with public IP addresses

--sc-additional-ips SC_ADDITIONAL_IPS
  SC interfaces that will be assigned
  with additional IP addresses

--mc-num-of-interfaces \{2,3,4\}
  number of interfaces for media components

--mc-public-ips MC_PUBLIC_IPS
  MC interfaces that will be assigned
  with public IP addresses

--mc-additional-ips MC_ADDITIONAL_IPS
  MC interfaces that will be assigned
  with additional IP addresses

--signaling1-subnet-id SIGNALING1_SUBNET_ID
  signaling 1 subnet id

--signaling2-subnet-id SIGNALING2_SUBNET_ID
  signaling 2 subnet id

--media1-subnet-id MEDIA1_SUBNET_ID
  media 1 subnet id

--media2-subnet-id MEDIA2_SUBNET_ID
  media 2 subnet id

--sc-instance-type SC_INSTANCE_TYPE
  signaling component instance type

--mc-instance-type MC_INSTANCE_TYPE
  media component instance type
The **modify** command is not allowed when some other operation is performed, for example, when the **scale-in** command is in progress.

```bash
$ stack_mgr modify stack1 --max-mc-num 5
Modifying stack configuration... done
```

The **modify** command has no effect on the stack service and completes without any delay. Some modifications require the **update** command to apply the changes. This is indicated in the **modify** command response:

```bash
$ stack_mgr modify stack1 --mc-num-of-interfaces 4
Modifying stack configuration... done
```

Stack configuration was modified.
Use 'update' command to apply the changes.

The indication is also provided in the output of the **show** command:

```bash
$ stack_mgr show stack1
<skipped>
Stack configuration changed           : update is needed
The following parameters were changed : mc_num_of_interfaces
```

### 4.9.1 Update Operation

The **update** command updates stack configuration. It is typically used after the **modify** command when the output of the latter indicates that an update is needed. For example, the **update** command is needed when the number of network interfaces on signaling or MCs is changed.

Note: The **Update** operation may be service affecting cause. It is therefore recommended to run it during periods of maintenance.

```bash
$ stack_mgr update --help
positional arguments:
    name         name of the stack

optional arguments:
    -h, --help    show this help message and exit
    -f, --force   force update even if it's not needed
```
Usually, the `update` command does nothing unless the 'update is needed' flag was turned on by the `modify` command. This behavior may be overridden by providing the '--force' argument.

```
$ stack_mgr update stack1
Initializing AWS client... done
Checking that configuration is allowed... done
Updating signaling components... done
Updating media components... done
Updating SBC cluster configuration... done
Wait for new configuration to be applied...... done
```

### 4.10 Stopping and Starting the Stack

#### 4.10.1 Stopping Stack

The `stop` command stops all stack components (both signaling and media). It is typically used to temporarily shut down stacks in a lab environment.

```
$ stack_mgr stop --help
usage: stack_mgr stop [-h] name

positional arguments:
  name  Defines the name of the stack

optional arguments:
  -h, --help  Show this help message and exit
  -i ids, --ids ids  comma-separated list of component id's, e.g. "mc-1,mc-3"
```

```
$ stack_mgr stop stack1
Initializing AWS client... done
Stopping stack components........................................ done
```

The `stop` command can also be used to stop specific components by using the `--ids` argument. This option is primarily used for debugging.

#### 4.10.2 Starting Stack

The `start` command starts all stack components. It is typically used after the `stop` command, to restore the stack to its operational state.

```
$ stack_mgr start --help
usage: stack_mgr start [-h] name

positional arguments:
  name  Defines the name of the stack

optional arguments:
```
The **start** command can also be used to start specific components by using the **--ids** argument. This option is primarily used for debugging.

### 4.11 Deleting Stack

The **delete** command deletes the existing stack. You must specify the stack name.

```
$ stack_mgr delete --help
usage: stack_mgr delete [-h] name

positional arguments:
  name        name of the stack

optional arguments:
  -h, --help  show this help message and exit
```

The **delete** process takes a few minutes and detailed progress information is displayed on the console:

```
$ stack_mgr delete stack1
Initializing AWS client... done
Deleting signaling components........................ done
Deleting media components............................... done
Deleting network resources.............................. done

Stack 'stack1' is successfully deleted
```

### 4.11.1 Purging Deleted Stack

The deleted stack is displayed by the **list** command (with the status "deleted") for 30 minutes after deletion:

```
$ stack_mgr list
+--------+-------------+-----+---------+
|  name  | type        | vim | state   |
+--------+-------------+-----+---------+
| stack1 | sbc-cluster | aws | deleted |
+--------+-------------+-----+---------+
```

If you want to immediately remove the deleted stack from the list, use the **purge** command:

```
$ stack_mgr purge stack1
Stack 'alex1' is purged
```
4.12 Healing Stack

The *Heal* operation verifies the state of all stack components and fixes any errors if detected. For example, it may remove MCs that are not properly registered in the SCs or remove orphaned entries from the "Media Components" configuration table.

```
$ stack_mgr heal stack1
Checking media components status... done
'mc-3' should be removed
Removing media components......................... done
```
4.13 Multiple Operations

Stack Manager limits every stack to a single operation (create, scale-out, scale-in, or update) at a time. Attempting to run some commands while other commands are in progress, results in the following output:

```
$ stack_mgr scale-out stack1
ERROR: stack 'stack1' is not in 'running' state (current state is 'scaling-in')
```

This limitation does not apply to the `show` and `list` commands, which can be performed in any state.

For different stacks, multiple operations can be performed simultaneously. For example, you can `scale-out stack1` while `stack2` is being `deleted`. 
5 REST API

5.1 Overview

The REST API is available under the /api/v1 path.

The following table provides a brief overview of the functionality supported using the REST API. Detailed information for each command is provided in subsequent sections.

<table>
<thead>
<tr>
<th>Method</th>
<th>Path</th>
<th>Command</th>
</tr>
</thead>
<tbody>
<tr>
<td>GET</td>
<td>/api/v1/stacks</td>
<td>list stacks</td>
</tr>
<tr>
<td>GET</td>
<td>/api/v1/stacks/&lt;stack_name&gt;</td>
<td>show stack</td>
</tr>
<tr>
<td>POST</td>
<td>/api/v1/stacks/&lt;stack_name&gt;</td>
<td>create stack</td>
</tr>
<tr>
<td>DELETE</td>
<td>/api/v1/stacks/&lt;stack_name&gt;</td>
<td>delete stack</td>
</tr>
<tr>
<td>PUT</td>
<td>/api/v1/stacks/&lt;stack_name&gt;</td>
<td>modify stack</td>
</tr>
<tr>
<td>PURGE</td>
<td>/api/v1/stacks/&lt;stack_name&gt;</td>
<td>purge stack</td>
</tr>
<tr>
<td>POST</td>
<td>/api/v1/stacks/&lt;stack_name&gt;/heal</td>
<td>heal stack</td>
</tr>
<tr>
<td>POST</td>
<td>/api/v1/stacks/&lt;stack_name&gt;/scale-in</td>
<td>scale-in stack</td>
</tr>
<tr>
<td>POST</td>
<td>/api/v1/stacks/&lt;stack_name&gt;/scale-out</td>
<td>scale-out stack</td>
</tr>
<tr>
<td>POST</td>
<td>/api/v1/stacks/&lt;stack_name&gt;/scale</td>
<td>scale stack</td>
</tr>
<tr>
<td>POST</td>
<td>/api/v1/stacks/&lt;stack_name&gt;/update</td>
<td>update stack</td>
</tr>
<tr>
<td>GET</td>
<td>/api/v1/config</td>
<td>get global configuration</td>
</tr>
<tr>
<td>PUT</td>
<td>/api/v1/config</td>
<td>update global configuration</td>
</tr>
</tbody>
</table>

5.2 Asynchronous Tasks

Most of the POST commands are performed asynchronously. A typical response contains a reference to an asynchronous task URL.

POST /api/v1/stack/<name>/scale-out

202 Accepted
Content-Type: application/json
{
    "description": "task accepted",
    "url": "/api/v1/tasks/1"
}

The REST client should poll this URL to get task status and detailed command output.

GET /api/v1/tasks/1

200 OK
Content-Type: application/json
{
    "status": "in_progress",
}
"output": "Removing SBC media components..."
}

GET /api/v1/tasks/1

200 OK
Content-Type: application/json
{
    "status": "success",
    "output": "Removing SBC media components....... done"
}

Valid task status values are:
- **idle** The task didn’t start execution yet
- **in_progress** The task is being executed
- **success** The task has successfully completed
- **failed** The task has failed

Note: The ‘output’ element may contain newline "\n" characters.

5.3 Authentication

Most of the REST API endpoints require basic HTTP authentication.
Use the same credentials (username/password) that are used for accessing the Web interface. Refer to Section 3.1, Accessing the Web Interface for more details.
5.4 Discovery

Method: GET
Path: /api/v1
Arguments: None
Description: Returns supported API structure

GET /api/v1

200 OK
Content-Type: application/json
{
  "items": [
    {
      "description": "list of available stacks",
      "id": "stacks",
      "url": "/api/v1/stacks"
    },
    {
      "description": "global configuration",
      "id": "config",
      "url": "/api/v1/config"
    },
    {
      "description": "application version",
      "id": "version",
      "url": "/api/v1/version"
    }
  ]
}

5.5 Global Configuration

Method: GET
Path: /api/v1/config
Arguments: None
Description: Returns Stack Manager's global configuration

GET /api/v1/config

200 OK
Content-Type: application/json
{
  "aws_access_key": "ABCDEFGHIJKLMNOPQRSTUVWXYZ",
  "aws_prefix": "",
  "aws_secret_key": "123456789012345678901234567890",
  "rest_api_password": "",
  "rest_api_username": ""
}
5.5.1 Updating Global Configuration

Method: PUT
Path: /api/v1/config
Arguments: None
Content Type: application/json
Content: Dictionary of parameter value/pairs
Description: Updates Stack Manager’s global configuration

PUT /api/v1/config
Content-Type: application/json
{
    "aws_access_key": "ABCDEDFGHIJKLMN",
    "aws_secret_key": "12345678901234567890"
}
200 OK
Content-Type: application/json
{
    "description": "success"
}

5.6 Listing Available Stacks

Method: GET
Path: /api/v1/stacks
Arguments: None
Description: Returns a list of all available stacks and basic information per stack

GET /api/v1/stacks
200 OK
Content-Type: application/json
{
    "stacks": [
        {
            "created_at": "Mar 14, 2018 16:59:15",
            "deleted_at": "",
            "id": "stack1",
            "management_ip": "51.124.138.162",
            "state": "running",
            "type": "sbc-cluster",
            "url": "/api/v1/stacks/alex1",
            "vim": "aws"
        }
    ]
}
5. REST API

5.7 Creating New Stack

Method: POST

Path: /api/v1/stacks/<stack_name>

Arguments: none

Content: configuration parameters as JSON dictionary
or
file – configuration file as multipart/form-data

Content type: application/json or multipart/form-data

Description: Creates new stack

Response: URL of asynchronous task (as described in 5.2)

POST /api/v1/stacks/stack1
Content-Type: application/json

{  
  "stack_type": "sbc-cluster",
  "vim": "aws",
  "mc_num": 3,
  "min_mc_num": 2,
  "max_mc_num": 10,
  "auto_scale": "enable",
  "media_util_scale_in_threshold": 250,
  "media_util_scale_out_threshold": 100,
  "dsp_util_scale_in_threshold": 0,
  "dsp_util_scale_out_threshold": 0,
  "auto_scale_cooldown_time": 900,
  "auto_scale_in_step": 1,
  "auto_scale_out_step": 1,
  "aws_region": "eu-central-1",
  "vpc_id": "vpc-45f3152c",
  "cluster_subnet_id": "subnet-be6e8bc3",
  "oam_subnet_id": "subnet-1536d368",
  "signaling1_subnet_id": "",
  "signaling2_subnet_id": "",
  "media1_subnet_id": "subnet-1536d368",
  "media2_subnet_id": "",
  "ssh_key_pair": "aws_ssh_frankfurt_1",
  "sc_ha_mode": "enable",
  "sc_num_of_interfaces": 2,
  "sc_public_ips": "eth1",
  "sc_additional_ips": "",
  "sc_image_id": "ami-d771563c",
  "sc_instance_type": "r4.2xlarge",
  "sc_iam_role": "SBC-HA-3",
  "sc_disk_size": 100,
  "sc_ini_file_contains_admin_user": "true",
  "sc_ini_file_url": "",
  "mc_num_of_interfaces": 3,
  "mc_public_ips": "eth1",
}
5.8 Checking Stack State and Configuration

Method: GET
Path: /api/v1/stacks/<stack_name>
Arguments: ?no_status=True Do not include real-time status information (connection status and media/dsp utilization) in the response.

Description: Returns detailed information of the specific stack. Unless ?no-status=True argument is provided, the command queries the active SC for real-time connection status and media/dsp utilization per MC. This may result in a delay in response (up to 30 seconds), for example, if connection with the active SC is unavailable.

GET /api/v1/stacks/stack1
200 OK
Content-Type: application/json
{
    "auto_scale": "enable",
    "auto_scale_cooldown_time": 900,
    "auto_scale_in_step": 1,
    "auto_scale_out_step": 1,
    "aws_region": "eu-central-1",
    "cluster_subnet_id": "subnet-be6e8bc3",
    "common_network_config": [  
        {
            "id": "subnet-be6e8bc3",
            "interface": "eth0",
            "status": "in-use",
            "subnet": "cluster"
        },
        {
            "id": "subnet-1536d368",
            "cron scheduler": "eth1",
            "status": "in-use",
            "subnet": "oam"
        }
    ],
    "connected_mc_num": 2,
    "created_at": "May 09, 2018 08:55:29",
    "deleted_at": "May 09, 2018 08:46:59",
    "dsp_util_scale_in_threshold": 0,
    "dsp_util_scale_out_threshold": 0,
    "free_dsp_resources": -1,
    "free_media_resources": 200,
    "id": "stack1",
    "manage_via_https": "enable",
    "management_ip": "18.197.127.204",
    "max_mc_num": 10,
    "mc_additional_ips": "",
    "mc_image_id": "ami-d771563c",
    "mc_instance_type": "r4.large",
    "mc_max_pps_limit": "auto",
    "mc_network_config": [  
        {
            "id": "subnet-1536d368",
            "interface": "eth2",
            "status": "in-use",
            "subnet": "media1"
        },
        {
            "id": "",
            "interface": "eth3",
            "status": "",
            "subnet": "media2"
        }
    ],
"mc_num": 2,
"mc_num_of_interfaces": 3,
"mc_profile": "forwarding",
"mc_public_ips": "eth1",
"mc_reboot_needed": false,
"mc_reboot_reason": "",
"medial1_subnet_id": "subnet-1536d368",
"media2_subnet_id": "",
"media_components": [
  {
    "created_at": "May 09, 2018 08:55:59",
    "dsp_util": -1,
    "id": "mc-1",
    "instance_type": "r4.large",
    "ip": "172.31.67.240",
    "media_util": 0,
    "status": "connected"
  },
  {
    "created_at": "May 09, 2018 08:55:59",
    "dsp_util": -1,
    "id": "mc-2",
    "instance_type": "r4.large",
    "ip": "172.31.67.15",
    "media_util": 0,
    "status": "connected"
  }
],
"media_util_scale_in_threshold": 250,
"media_util_scale_out_threshold": 100,
"min_mc_num": 2,
"name_prefix": "",
"oam_subnet_id": "subnet-1536d368",
"sc_additional_ips": "",
"sc_disk_size": 100,
"sc_ha_mode": "enable",
"sc_iam_role": "SBC-HA-3",
"sc_image_id": "ami-d771563c",
"sc_ini_file_contains_admin_user": "true",
"sc_ini_file_url": "",
"sc_instance_type": "r4.2xlarge",
"sc_network_config": [
  {
    "id": "",
    "interface": "eth2",
    "status": "",
    "subnet": "signaling1"
  },
  {
    "id": "",
    "interface": "eth3",
    "status": "",
    "subnet": "signaling1"
  }
]
"status": "",
"subnet": "signaling2"
],
"sc_num_of_interfaces": 2,
"sc_public_ips": "eth1",
"sc_reboot_needed": false,
"sc_reboot_reason": "",
"signaling1_subnet_id": "",
"signaling2_subnet_id": "",
"signaling_components": [
{
"created_at": "May 09, 2018 08:57:19",
"id": "sc-1",
"instance_type": "r4.2xlarge",
"ip": "172.31.71.211",
"status": "running"
},
{
"created_at": "May 09, 2018 08:57:19",
"id": "sc-2",
"instance_type": "r4.2xlarge",
"ip": "",
"status": "running"
}
],
"ssh_key_pair": "aws_ssh_frankfurt_1",
"stack_type": "sbc-cluster",
"state": "idle",
"update_needed": false,
"vim": "aws",
"vpc_id": "vpc-45f3152c"}
5.9 Scaling Mediant CE Stack

5.9.1 Scale Out Operation

Method: POST
Path: /api/v1/stacks/<stack_name>/scale-out
Arguments:
?num=2 Defines the number of MCs to add
Description: Scales out the stack
Response: URL of asynchronous task (as described in Section 5.2)

```
POST /api/v1/stacks/stack1/scale-out

202 Accepted
Content-Type: application/json
{
  "description": "task accepted",
  "url": "/api/v1/tasks/1"
}
```

5.9.2 Scale In Operation

Method: POST
Path: /api/v1/stacks/<stack_name>/scale-in
Arguments:
?num=2 Defines the number of MCs to remove
?ids=mc-1,mc-2 Comma-separated list of IDs of MCs to remove
Description: Scales in the stack
Response: URL of asynchronous task (as described in 5.2)

```
POST /api/v1/stacks/stack1/scale-in

202 Accepted
Content-Type: application/json
{
  "description": "task accepted",
  "url": "/api/v1/tasks/1"
}
```
### 5.9.3 Scale To Operation

**Method:** POST  
**Path:** /api/v1/stacks/<stack_name>/scale  
**Arguments:**  
- `?num=2` Defines the number of MCs  
**Description:** Scales the stack to the specified number of MCs  
**Response:** URL of asynchronous task (as described in Section 5.2)

```plaintext
POST /api/v1/stacks/stack1/scale?num=2

202 Accepted
Content-Type: application/json
{
    "description": "task accepted",
    "url": "/api/v1/tasks/1"
}
```
## 5.10 Modifying Stack Configuration

<table>
<thead>
<tr>
<th>Method</th>
<th>PUT</th>
</tr>
</thead>
<tbody>
<tr>
<td>Path</td>
<td>/api/v1/stacks/&lt;stack_name&gt;</td>
</tr>
<tr>
<td>Arguments</td>
<td>None</td>
</tr>
<tr>
<td>Content type</td>
<td>application/json</td>
</tr>
<tr>
<td>Content</td>
<td>Dictionary of parameter value/pairs</td>
</tr>
</tbody>
</table>

**Description:**
Modifies the stack configuration

```plaintext
PUT /api/v1/stacks/stack1
Content-Type: application/json

{   "auto_scale ": "enable",   "media_util_scale_in_threshold": 230}
```

200 OK

```plaintext
Content-Type: application/json

{   "description": "stack configuration was modified"
}
```

Some modify actions require stack updates to be run to apply them. This is indicated using the `update_needed` attribute in the response. The `update_needed` flag is set on the stack.

```plaintext
PUT /api/v1/stacks/stack1
Content-Type: application/json

{   "max_mc_num": 10}
```

200 OK

```plaintext
Content-Type: application/json

{   "description": "stack configuration was modified; stack must be updated to apply the changes",   "update_needed": True,   "url": "/api/v1/stacks/stack1/update"
}
```
5.10.1 Update Operation

Method: POST
Path: /api/v1/stacks/<stack_name>/update
Arguments:
  ?force=True Forces update even if it's not needed
  ?reset=True Resets 'update is needed' flag without performing the update

Description: Updates the stack
Response: URL of asynchronous task (as described in Section 5.2)

POST /api/v1/stacks/stack1/update

202 Accepted
Content-Type: application/json
{
    "description": "task accepted",
    "url": "/api/v1/tasks/1"
}
5.11 Stopping and Starting Stack

5.11.1 Stopping Stack

Method: POST  
Path: /api/v1/stacks/<stack_name>/stop  
Arguments:  
?ids=mc-1,mc-3  
Description: Stops stack components  
Response: URL of asynchronous task (as described in 5.2)

POST /api/v1/stacks/stack1/stop

202 Accepted  
Content-Type: application/json
{
    "description": "task accepted",
    "url": "/api/v1/tasks/1"
}

5.11.2 Starting Stack

Method: POST  
Path: /api/v1/stacks/<stack_name>/start  
Arguments:  
?ids=mc-1,mc-3  
Description: Starts stack components  
Response: URL of asynchronous task (as described in 5.2)

POST /api/v1/stacks/stack1/start

202 Accepted  
Content-Type: application/json
{
    "description": "task accepted",
    "url": "/api/v1/tasks/1"
}
5.12 Deleting Stack

Method: DELETE
Path: /api/v1/stacks/<stack_name>
Arguments: none
Description: Deletes stack
Response: URL of asynchronous task (as described in 5.2)

```
DELETE /api/v1/stacks/stack1

202 Accepted
Content-Type: application/json
{
   "description": "task accepted",
   "url": "/api/v1/tasks/1"
}
```

5.12.1 Purging Deleted Stack

Method: PURGE
Path: /api/v1/stacks/<stack_name>
Arguments: none
Description: Purges deleted stack

```
PURGE /api/v1/stacks/stack1

200 OK
Content-Type: application/json
{
   "description": "stack was purged"
}
```
5.13 Healing Stack

Method: POST
Path: /api/v1/stacks/<stack_name>/heal
Arguments: none
Description: Heals the stack
Response: URL of asynchronous task (as described in Section 5.2)

```
POST /api/v1/stacks/stack1/heal

202 Accepted
Content-Type: application/json

{
    "description": "task accepted",
    "url": "/api/v1/tasks/1"
}
```
Stack Manager stores its logs in the `/var/log/stack_mgr` directory. The following files are created:

- `stack_mgr.log`: Application log file
- `auto_scale.log`: Automatic scaling logs
- `auto_heal.log`: Automatic healing logs
- `rest_api.log`: HTTP server logs
- `rest_api_access.log`: HTTP server access log
- `rest_api_error.log`: HTTP server error log

Log files are rotated daily. Up to seven copies of each file are stored.

To view logs through the Web interface, open the Logs page, and then choose the corresponding log.

To view logs through CLI, use regular Linux commands, for example, `tail` or `less`:

```bash
$ tail -f /var/log/stack_mgr/stack_mgr.log
```