# Oracle® Communications WebRTC Session Controller

Installation Guide

Release 7.2

**E69510-02**

April 2017

ORACLE®

Oracle Communications WebRTC Session Controller Installation Guide, Release 7.2

E69510-02

# Contents

## 4 WebRTC Session Controller Pre-Installation Tasks



## 5 Installing WebRTC Session Controller Signaling Engine



## 6 Creating and Configuring a WebRTC Session Controller Signaling Engine Domain



## 7 WebRTC Session Controller Signaling Engine Post-Installation Tasks



## 8 Upgrading WebRTC Session Controller Signaling Engine from an Earlier Version

## 9 WebRTC Session Controller Media Engine Installation Overview



## 10 Quick Commissioning New Media Engine Systems

## 11 Installing Media Engine Clusters

## 12 Configuring Secure Media (SRTP) Sessions



## 13 Creating and Commissioning USB Sticks



## 14 Installing and Running the ME Virtual Machine



## 15 Upgrading WebRTC Session Controller Media Engine From an Earlier Version



## 16 Troubleshooting a WebRTC Session Controller Installation

# Preface

This guide describes the system requirements and procedures for installing Oracle Communications WebRTC Session Controller.

## Audience

This document is for system administrators who install and configure the WebRTC Session Controller. The person installing the software should be familiar with the following topics:

- Operating system commands
- Network Management
- Oracle Coherence

Before reading this guide, you should familiarize yourself with WebRTC Session Controller. See *Oracle Communications WebRTC Session Controller Concepts*.

## Documentation Accessibility

For information about Oracle's commitment to accessibility, visit the Oracle Accessibility Program website at http://www.oracle.com/pls/topic/lookup?ctx=acc&id=docacc.

### Access to Oracle Support

Oracle customers that have purchased support have access to electronic support through My Oracle Support. For information, visit http://www.oracle.com/pls/topic/lookup?ctx=acc&id=info or visit http://www.oracle.com/pls/topic/lookup?ctx=acc&id=trs if you are hearing impaired.

## Accessing Oracle Communications Documentation

WebRTC Session Controller documentation is available from the Oracle Documentation Web site: http://docs.oracle.com.

## Related Documents

Refer to these additional documents for related information on WebRTC Session Controller:

- *Oracle Communications WebRTC Session Controller Concepts*

- *Oracle Communications WebRTC Session Controller Extension Developer's Guide*
- *Oracle Communications WebRTC Session Controller System Administrator's Guide*
- *Oracle Communications WebRTC Session Controller Security Guide*
- *Oracle Communications WebRTC Session Controller Media Engine Object Reference*
- *Oracle Communications WebRTC Session Controller Release Notes*
- *Oracle Fusion Middleware 12c Documentation Library*

# 1

# WebRTC Session Controller Installation Overview

This chapter provides an overview of Oracle Communications WebRTC Session Controller installed components and of the WebRTC Session Controller installation process. Subsequent chapters describe installation steps in detail.

For more detailed WebRTC Session Controller overview information, see the discussion about system architecture in *WebRTC Session Controller Concepts*.

## About WebRTC Session Controller Media Engine and Signaling Engine

WebRTC Session Controller has two main sub components:

- WebRTC Session Controller Signaling Engine (Signaling Engine)
- WebRTC Session Controller Media Engine (Media Engine)

Signaling Engine handles signaling services between Web browser clients, traditional Session Initialization Protocol (SIP) networks and the Media Engine, while Media Engine handles media streaming as well as Traversal Using Relay Network Address Translation (TURN) functionality.

Although these two sub components work together as a single solution, and cannot be used independently, they employ completely different installation models. For instance, Signaling Engine is currently certified on a single operating system, while Media Engine supports a selection of bare metal servers. Likewise, Signaling Engine utilizes a graphical installer, while Media Engine is a command line driven installation. In addition, each component requires additional independent configuration and post installation steps. For those reasons, you must pay careful attention to the instructions contained in this guide.

## Overview of the WebRTC Session Controller Installation Procedure

The installation procedure follows these steps:

1. Plan your installation. When planning your installation, you do the following:
    - Determine the scale of your implementation, for example, a small test system or a large production system.
    - Determine how many physical machines you need, and which software components to install on each machine.
    - Plan the system topology, for example, how the system components connect to each other over the network.

See "Planning Your WebRTC Session Controller Installation" for more information.

2. Review system requirements. System requirements include:
   - Hardware requirements, such as disk space
   - System software requirements, such as operating system (OS) versions and OS patch requirements, and Java Virtual Machine (JVM) process requirements (such as memory settings)
   - Information requirements, such as IP addresses and host names

   See "WebRTC Session Controller System Requirements" for more information.
3. Perform pre-installation tasks including assigning IP addresses to network assets and installing necessary support software.

   See "WebRTC Session Controller Pre-Installation Tasks" for more information.
4. Install WebRTC Session Controller Signaling Engine.

   See "Installing WebRTC Session Controller Signaling Engine" for more information.
5. Configure a WebRTC Session Controller Signaling Engine domain.

   See "Creating and Configuring a WebRTC Session Controller Signaling Engine Domain" for more information.
6. Perform Signaling Engine post-installation tasks.

   See "WebRTC Session Controller Signaling Engine Post-Installation Tasks" for more information.
7. Install WebRTC Session Controller Media Engine.

   See "WebRTC Session Controller Media Engine Installation Overview" for more information.
8. Troubleshoot any installation issues.

   See "Troubleshooting a WebRTC Session Controller Installation" for more information.

## Ensuring a Successful Installation

The WebRTC Session Controller installation should be performed by qualified personnel. You must be familiar with both Signaling Engine and Media Engine software and the operating systems on which you are installing the software.

Follow these guidelines:

- As you install each component; for example, the JDK and WebRTC Session Controller Signaling Engine, verify that the component installed successfully before continuing the installation process.
- Pay close attention to the system requirements. Before you begin installing the software, make sure your system has the required base software. In addition, make sure that you know all of the required configuration values, such as host names and port numbers.
- As you create new configuration values, write them down. In some cases, you might need to re-enter configuration values later in the procedure.

## Planning the Network and Hardware Setup

Before you can install WebRTC Session Controller, you must gather some information about your system and decide on the directories in which to install the software. You need to know:

- The network names or IP addresses of the machines on which you are going to install the following components:
  - The Signaling Engine
  - The Media Engine
- For machines hosting Signaling Engine installations, the directory on each machine which will serve as your *Middleware_home* directory. This directory serves as a repository for common files that are used by WebRTC Session Controller products installed on the same machine, such as WebLogic Server and a Java Development Kit.

  The files in the *Middleware_home* directory are essential to ensuring that software operates correctly on your system. They:
  - Facilitate checking of cross-product dependencies during installation
  - Facilitate Service Pack installation
- Passwords for administrative users on all server types.

WebRTC Session Controller has been tested to run on specific hardware and software platforms. "WebRTC Session Controller System Requirements" outlines supported configurations in detail. Unless your installation has been specified differently in cooperation with Oracle, only those configurations are supported.

## Directory Placeholders Used in This Guide

Table 1–1 lists placeholders that are used in this guide to refer to the directories that contain WebRTC Session Controller system components.

***Table 1–1 Directory Placeholders***

| Placeholder | Directory |
|---|---|
| *Central_inventory_location* | The directory in which the Oracle inventory file lives. The Oracle inventory lists all Oracle software installed on a machine. Created by the Oracle installer upon the first Oracle product installation. |
| *Domain_home* | The location of a configured WebRTC Session Controller WebLogic domain. Created by the Signaling Engine domain creation wizard after product installation. |
| *Middleware_home* | The top level location for Oracle product files. Created during product installation. |
| *Oracle_home* | The storage location for common Oracle software files within *Middleware_home*. Created during product installation. |
| *temp_dir* | A temporary directory into which you extract the installation images for Signaling and Media engine. Referenced when extracting installation files and other utilities. |
| *WSC_home* | The directory in which WebRTC Session Controller is installed. Created during product installation. |

# 2

# Planning Your WebRTC Session Controller Installation

This chapter provides information about planning your Oracle Communications WebRTC Session Controller installation.

## About Planning Your WebRTC Session Controller Installation

When planning a WebRTC Session Controller installation, you consider how many physical servers can handle your subscriber base and how many signaling and media server nodes to include in your cluster.

## About Development Systems, and Production Systems

A WebRTC Session Controller development system and a WebRTC Session Controller production system may differ based on the number of elements such as the machines, security levels and configuration mode.

See "WebRTC Session Controller System Requirements" for information about required hardware and software.

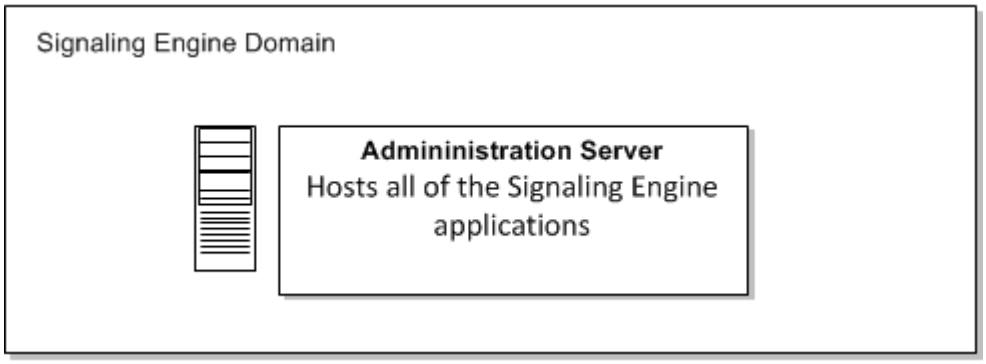
## Planning Your Signaling Engine Installation

The following section describes recommended Signaling Engine installation topologies.

### Understanding Signaling Engine Installation Topologies

Figure 2–1 shows simple Signaling Engine installation topology, a domain with a single Administration server. The Administration server hosts all of the Signaling Engine applications on a single machine.

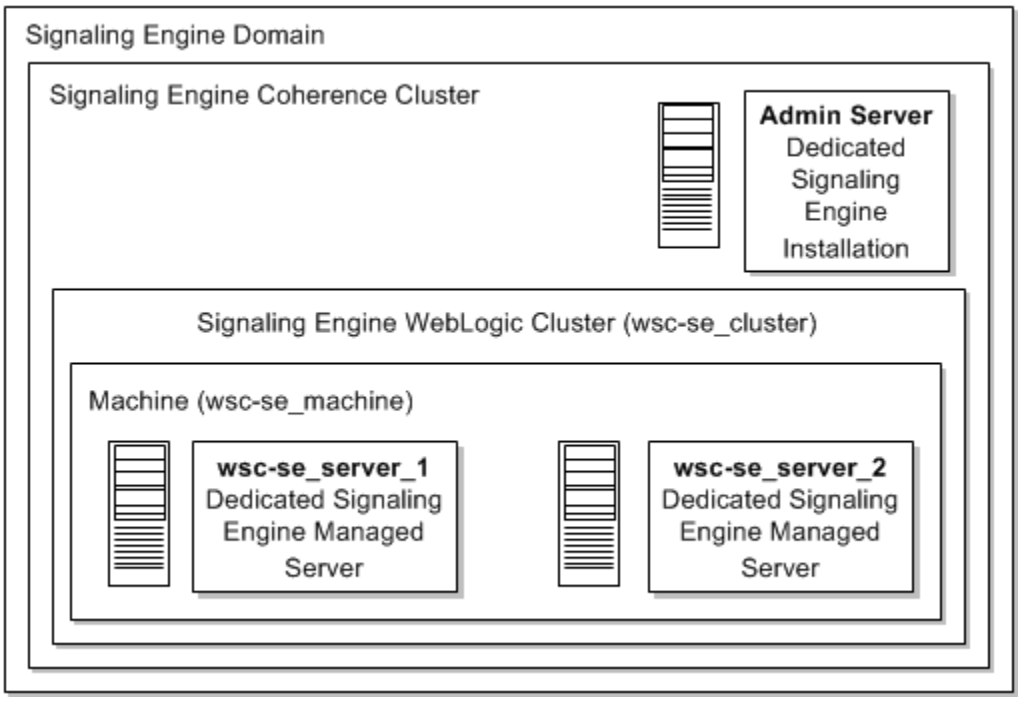
***Figure 2–1 Basic Single Server Signaling Engine Domain***




While a simple single host configuration is sufficient for development and Proof of Concept (PoC) installations, for production systems, a robust, fault-tolerant system is required.

Figure 2–2 shows a fault tolerant Signaling Engine installation topology. In this topology, the Administration server is installed on a separate machine from the two independently hosted clustered Managed Servers. The Managed Servers can be separated geographically for additional fault tolerance, and additional engines can be added as required.

***Figure 2–2 Fault Tolerant Signaling Engine Domain***




Each element in this topology illustration is described in Table 2–1.

***Table 2–1 Description of the Elements in the Signaling Engine Server and Coherence Standard Installation Topology***

| Element | Description and Links to Additional Documentation |
|---|---|
| Signaling Engine Domain | A logically related group of Java components (in this case, the Administration Server, Managed Signaling Engines, and other related software components).<br>For more information, see "Understanding Domains" in *Understanding Oracle WebLogic Server*. |
| Administration Server | The central control entity of a domain which maintains the domain's configuration objects and distributes configuration changes to Managed Servers.<br>For more information, see "Administration Server" in *Understanding Oracle WebLogic Server*. |
| Cluster | A collection of multiple Signaling Engine instances running simultaneously and working together.<br>For more information, see "Managed Servers and Managed Server Clusters" in *Understanding Oracle WebLogic Server*. |
| Machine | Logical representation of the computer that hosts one or more WebLogic Server instances (servers). Machines are also the logical glue between Signaling Engine Managed Servers and the Node Manager; in order to start or stop a Managed Server with Node Manager, the Managed Server must be associated with a machine. |
| Managed Server | Host for your applications, application components, Web services, and their associated resources.<br>For more information, see "Managed Servers and Managed Server Clusters" in *Understanding Oracle WebLogic Server*. |

### WebRTC Session Controller Signaling Engine Coherence Planning

WebRTC Session Controller Signaling Engine nodes are based on Oracle Coherence. Decide how to configure Oracle Coherence settings for your WebRTC Session Controller Signaling Engine topology, for example, how many nodes to add to the cluster when a node failure occurs. For more information, see "Configuring and Managing Coherence Clusters" in *Administering Clusters for Oracle WebLogic Server*.

## Planning Your Media Engine Installation

Media Engine nodes are installed on certified bare metal servers. Additional nodes can be added as required to support greater volumes of media traffic. For more information see "WebRTC Session Controller Media Engine Installation Overview".

## About Installing a Secure System

In a production system, you must ensure that communication between components and access to the system servers are secure. For information about choices for installing a secure system, see *Oracle Communications WebRTC Session Controller Security Guide*.

# 3

# WebRTC Session Controller System Requirements

This chapter describes the software, hardware, and information requirements for Oracle Communications WebRTC Session Controller.

## Software Requirements

This section describes the required software for the two WebRTC Session Controller sub components, Signaling Engine and Media Engine.

### Signaling Engine Software Requirements

Signaling Engine is certified on Oracle Linux x64 versions 6 and 7, running either natively or as a part of Oracle VM Server.

In addition, Signaling Engine requires a 64-bit Java Development Kit (JDK) version 1.8 plus the latest security update.

**Note:** OpenJDKs of any version are not supported.

### Signaling Engine Hardware Requirements

The following platforms have been certified for use with the Signaling Engine:

- Kernel-based Virtual Machine (KVM)
- VMware ESXi
- Oracle Virtual Machine (OVM)/Oracle Enterprise Linux (OEL)

### Media Engine Software Requirements

Media Engine is certified to run on Oracle Linux version 7.0 or higher and uses yum to install and update RPM files. For more information, see "WebRTC Session Controller Media Engine Installation Overview".

### Media Engine Hardware Requirements

The following platforms have been certified for use with the Media Engine:

- Sun Netra X5-2
- Sun Server X5-2

- Sun Netra X3-2
- HP DL160 G9
- NN2610
- NN2620

The following VM platforms have been certified for use with the Media Engine:

- OVM 3.3.1
- VMware ESXi 5.5
- Xen 3.4.3
- KVM on OL7

## Port Requirements

WebRTC Session Controller requires access to the following port types:

- **WebRTC client ports**: Ports that WebRTC applications use to communicate with WebRTC Session Controller. Client ports must be exposed through a firewall to client applications.
- **SIP network ports**: Ports that WebRTC Session Controller uses to communicate with the SIP network. The SIP network is usually internal.
- **Media ports**: Ports used for media anchoring; exposed to both WebRTC clients and the SIP network. Media ports must be exposed through a firewall to client applications.
- **Internal administration ports**: Ports used for administration of WebRTC Session Controller. Internal administration ports need not be exposed externally, but must be accessible between Signalling Engine and Media Engine instances.

Table 3–1 lists the Signalling Engine port requirements.

***Table 3–1 Signalling Engine Port Requirements***

| Port | Port Type | Description |
|---|---|---|
| 80 | **WebRTC client port** | The default port for WebRTC client communication (TCP). This port is not required if you are using Websockets Secure (WSS), 443. |
| 443 | **WebRTC client port** | The port for WebRTC client communication if you are using WSS (TCP). If you are using WSS you do not need port 80 open. |
| 4057 | **Internal administration port** | WebRTC Session Controller Media Engine HTTP callback port. Used for Signalling Engine/Media Engine communications. |
| 5060 | **SIP network port** | SIP network port. The default Session Initiation Protocol (SIP) port used to communicate with the SIP network. Not required if you are using Secure SIP (SIPS). |
| 5061 | **SIP network port** | The default Secure SIP (SIPS) port used to communicate with the SIP network. Not required if you are using the regular SIP port. |
| 7001 | **Internal administration port** | The default Signalling Engine Administration HTTP port. Not required if you are using the Secure Sockets Layer (SSL) administration port, 7002. |

*Table 3–1 (Cont.) Signalling Engine Port Requirements*

| Port | Port Type | Description |
|---|---|---|
| 7002 | **Internal administration port** | The default SSL Signalling Engine Administration HTTPS port. Not required if you are using the non-SSL port. |
| 8443 | **WebRTC client port** | The port for communication between the Signaling Engine and the Media Engine. |

Table 3–2 lists the Media Engine port requirements.

*Table 3–2 Media Engine Port Requirements*

| Port | Port Type | Description |
|---|---|---|
| 8080 | Internal administration port | The Media Engine load factor application port, used for Signalling Engine/Media Engine communications. The load factor application reports to the Signalling Engine on the status of connected Media Engine instances. |
| Media port range (default 20000-24999) | **Media ports** | The UDP Media Engine media anchoring ports (SRTP and DTLS) |
| 3478 | **WebRTC client port** | The Traversal Using Relays around NAT (TURN) or Session Traversal Utilities for NAT (STUN) port (UDP, TCP, and TLS). |

## About Critical Patch Updates

WebRTC Session Controller is supported on all Oracle Critical Patch Updates. You should install all Critical Patch Updates as soon as possible.

To download Critical Patch Updates, find out about security alerts, and enable email notifications about Critical Patch Updates, see the Security topic on Oracle Technology Network:

http://www.oracle.com/technetwork/topics/security/whatsnew/index.html

# Hardware Requirements

This section describes the required hardware for the two WebRTC Session Controller sub components, Signaling Engine and Media Engine.

## Signaling Engine Hardware Requirements

The number and configuration of the computers that you employ for your Signaling Engine installation depend on the scale and the kind of deployment you have planned according to your charging requirements. You will need to work with your performance team to determine your sizing requirements.

Signaling Engine has similar requirements to Oracle WebLogic Server 12c. The following items are required in addition to the basic WebLogic Server requirements:

- Gigabit Ethernet connections are required between engine servers for all production deployments.
- Dual network interface cards (NICs) are required to provide fail-over capabilities in a production environment.

- Additional RAM is required to support the throughput requirements of most production installations.

> **Note:** Each Transport Control Protocol (TCP) WebSocket connection requires approximately 14 kilobytes of RAM.

## Media Engine Hardware Requirements

Media Engine is certified to run on several hardware platforms. For a complete list, see "Media Engine Hardware Requirements".

While Media Engine may run on other configurations, you are likely to run into disk controller as well as networking controller issues.

The number of physical or virtual servers will depend upon your particular environment load, but at a minimum each Media Engine server requires:

- Gigabit Ethernet connections
- 4 GB of RAM
- At least 50 GB of free hard disk space
- 64-bit Intel processor with two CPU cores

# 4

# WebRTC Session Controller Pre-Installation Tasks

This chapter describes pre-installation tasks for Oracle Communications WebRTC Session Controller.

## About Pre-Installation Tasks

You must perform certain tasks before installing WebRTC Session Controller. Pre-installation tasks are broken down into the following categories:

- General Pre-Installation Tasks: Tasks that are not specific to either the WebRTC Session Controller Media Engine (Media Engine) or WebRTC Session Controller Signaling Engine (Signaling Engine) components but that are required for a functioning installation.
- Signaling Engine Pre-Installation Tasks: Tasks that you must perform before you install Signaling Engine.

## General Pre-Installation Tasks

Before continuing, you must complete the following general pre-installation tasks:

- You should allocate IP addresses for Media Engine and Signaling Engine interfaces, including:
  - Public-facing external interfaces
  - Internal-facing interfaces
  - Intra-system interfaces for private signaling between Media Engine and Signaling engine nodes
  - If required by your organization, a separate systems management interface
- You should determine which logical interfaces map to which physical interfaces on each server.
- You should have access to a Domain Name Service (DNS) and Network Time Protocol (NTP) servers.
- Optionally you can assign fully qualified domain names to each server.
- You should make a note of any required static routes.
- You should download the WebRTC Session Controller software.

  To obtain the WebRTC Session Controller Signaling Engine Software:

1. Download the WebRTC Session Controller software from the Oracle software delivery Web site, located at:

   http://edelivery.oracle.com

   and save it to a temporary directory (*temp_dir*).
2. Unzip the WebRTC Session Controller installation files.

- You should download the WebRTC Media Engine software and copy it to a USB stick.

  Software can be downloaded from either the Oracle Software Delivery Cloud or the My Oracle Support Patches and Updates tab.

  To access the Oracle Software Delivery Cloud:

  1. Access the https://edelivery.oracle.com link.
  2. Select the **Sign In/Register** tab and enter your **username** and **password**.

     **Note:** If you are a new user, you must create an account.

  3. Click the checkbox to agree to the to the **Oracle Trial License Agreement and Export Restrictions** and click **Continue**.
  4. Select the Oracle Communications product pack.
  5. Select the Acme Packet OS platform and click **Go**.
  6. Download the following file and click **Download**.
     - Oracle Communications WebRTC Session Controller 7.2 Installation Repository
  7. Copy the file onto a USB stick.

  To access the Oracle Support Software Patches and Updates:

  1. Log into the My Oracle Support Portal.
  2. Select the **Patches and Updates** tab.
  3. Select the **Search** tab and click **Product or Family (Advanced)**.
  4. **Product:** Enter **Oracle Communications Application Session Controller**.
  5. **Release:** Enter **WebRTC Session Controller 7.2**.
  6. Click **Search**. The available distribution formats appear and include the following information:
     - Patch Name
     - Description
     - Release
     - Platform (Language)
     - Classification
     - Product
     - Prerequisite Requirement
     - Size

– Download Access

7. Select the distribution format that you require.
8. Click either **Download** to download the file or **Read Me** to view the Build Notes for this patch.
9. Copy the file onto a USB stick.

# Signaling Engine Pre-Installation Tasks

Before installing Signaling Engine software, you must complete the following pre-installation tasks:

- Install a Java Development Kit
- Create the Signaling Engine User Account
- Choose an Installation Directory

## Install a Java Development Kit

Install an Oracle Java Development Kit (JDK) 1.8 plus the latest security update, and add it to your PATH environment variable. The JRE is required for the installer process.

You can download a JDK from the Java SE Development Kit 8 Downloads page: `http://www.oracle.com/technetwork/java/javase/downloads/jdk8-downloads-2133151.html`.

See "JDK 8 Installation for Linux Platforms" in the Oracle Java SE Documentation for instructions on installing a JDK.

---

**Note:** Signaling Engine is not compatible with the OpenJDK installed by default in development installations of Oracle Linux 6, nor is it compatible with Oracle JDK 1.7.0_40.

---

## Create the Signaling Engine User Account

Create the user account that is to be the primary user running Signaling Engine in your environment.

It is required that all machines have the same user name configured.

Note the user name; you will be required to specify the user name during the Signaling Engine installation process.

To create the user account:

1. Log in to the driver machine.
2. Enter the following command:

   **useradd** *passwd user_name*

   where *user_name* is the name of the user and *passwd* is the password for the user.

See the discussion in your Linux documentation for more information about the **useradd** command.

### Choose an Installation Directory

When you install WebRTC Session Controller, you are prompted to specify a Middleware home directory. This directory serves as a repository for common files that are used by multiple Fusion Middleware products installed on the same machine. For this reason, the Middleware home directory can be considered a central support directory for all the Fusion Middleware products installed on your system.

The files in the Middleware home directory are essential to ensuring that WebRTC Session Controller and WebLogic Server operate correctly on your system. They facilitate checking of cross-product dependencies during installation.

For more information on choosing an installation directory, see "Understanding the Oracle WebLogic Server and Coherence Directory Structure" in *Installing and Configuring Oracle WebLogic Server and Coherence*.

## Next Steps

After you have completed pre-installation tasks you can install Signaling Engine. See "Installing WebRTC Session Controller Signaling Engine" for instructions.

# 5

# Installing WebRTC Session Controller Signaling Engine

This chapter describes how to install Oracle Communications WebRTC Session Controller Signaling Engine (Signaling Engine).

> **Note:** If you are installing a patch set for the Signaling Engine, refer to the *Release Notes* and the *ReadMe* files that accompany the patch set for information on installing it.

Before you install Signaling Engine, you must complete all pre-installation tasks described in "WebRTC Session Controller Pre-Installation Tasks".

## About the GUI Installation and Silent Installation

You can install Signaling Engine by using the GUI installation or the silent installation. The silent installation procedure enables you to perform a non-interactive installation of Signaling Engine. You can use the silent installation to install Signaling Engine quickly on multiple systems.

The silent installer uses a response file in which you specify installation settings. To obtain the response file, you first run the GUI installation, and choose to save a response file.

For installation instructions, see the following sections:

- Installing Signaling Engine Using the GUI Installation
- Installing Signaling Engine Using the Silent Installation

## Installing Signaling Engine Using the GUI Installation

To install Signaling Engine:

1. If you have not done so already, download and unzip the WebRTC Session Controller software. See "General Pre-Installation Tasks" for instructions.
2. Log in to the system on which you want to install Signaling Engine.
3. The installer requires that a certified JDK already exists on your system. For more information, see "Install a Java Development Kit".
4. Go to the directory where you downloaded the installation program.

5. Launch the installation program by invoking `java -jar` from the JDK directory on your system, as shown in the example below:

   ```
   java -jar wsc_generic.jar
   ```

   If no other Oracle products are installed on the system, the Installation Inventory screen appears. Specify the location where you want to create your central inventory. Make sure that the operating system group name selected on this screen has write permissions to the central inventory location.

   For more information about the central inventory, see "Understanding the Oracle Central Inventory" in *Installing Software with the Oracle Universal Installer*.

   If an Installation Inventory already exists, the WebRTC Session Controller Installation Welcome window appears.

6. Click **Next**.

   The WebRTC Session Controller Oracle Installation Location window appears.

   Use this screen to specify the location of your Oracle home directory.

   For more information about Oracle Fusion Middleware directory structure, see "Selecting Directories for Installation and Configuration" in *Planning an Installation of Oracle Fusion Middleware*.

7. Click **Next**.

   The Installation Type window appears.

   You can choose either "**WebRTC Session Controller Installation**" which includes only the WebRTC Session Controller software or "**Complete Installation**" which also includes WebRTC Session Controller sample applications.

   For more information on the WebRTC Session Controller sample applications, see *Oracle Communications WebRTC Session Controller Application Developer's Guide*.

8. Click **Next**.

   The Prerequisite Checks window appears.

   This screen verifies that your system meets the minimum necessary requirements.

   If there are any warnings or errors, make sure that your environment meets all the necessary prerequisites. See "WebRTC Session Controller System Requirements" for more information.

9. Click **Next**.

   The Security Updates window appears.

   If you already have an Oracle Support account, use this screen to indicate how you would like to receive security updates.

   If you do not have one and are sure you want to skip this step, clear the check box and verify your selection in the follow-up dialog box.

10. Click **Next**.

    The Installation Options screen appears.

    Use this screen to verify the installation options you selected. If you want to save these options to a response file, click **Save Response File** and provide a location and name for the response file. Response files can be used later in a silent installation situation.

**Note:** If you only want to save a response file, you can exit the installation at this time.

For more information about silent mode installation, see "Installing Signaling Engine Using the Silent Installation".

11. Review the selections you have made, and click **Install**.

    The Installation Progress window appears and the installation begins.

    **Note:** After the installation begins, if you click **Cancel**, the installation process stops but the files that are already copied are not removed.

12. When the installation is completed, click **Next**.

    The Installation Complete window appears.

    The next step is to launch the configuration wizard to create your WebLogic domain. There are two ways to do this:

    - Select **Automatically Launch the Configuration Wizard** on this screen. After you click **Finish** to close the installer, the configuration wizard is started and you can begin to configure your domain. If you choose to do this, proceed to "Configuring a WebRTC Session Controller Domain".
    - Do not select **Automatically Launch the Configuration Wizard** on this screen. After you click **Finish** to close the installer, you must manually start the configuration wizard to begin configuring your domain. If you choose to do this, proceed to "Starting the Configuration Wizard".

# Installing Signaling Engine Using the Silent Installation

The silent installation uses a response file in which you have set installation information. To obtain the response file, you run the GUI installer up until the Installation Options pane appears. You can then save a response file that contains the key and values pairs based on the values that you specify during the GUI installation. You can then copy and edit the response file to create additional response files for installing Signaling Engine on additional machines.

## Creating a Response File

The response file must contain the key and value pairs for the mandatory parameters the installer must use for the Signaling software component you install. All information prompted in the GUI installation is associated with mandatory parameters.

To create a response file:

1. Run the GUI installation for Signaling Engine. See "Installing Signaling Engine Using the GUI Installation" for instructions.
2. When the Installation Options screen appears, click **Save Response File** and provide the location and name for the response file.
3. Modify the response file you copied by specifying the key and value information for the parameters you want in your installation.

4. Save and close the response file.

> **Note:** For information on response file key and value pairs, see "Using the Oracle Universal Installer in Silent Mode" in *Installing Software with the Oracle Universal Installer*.

## Performing a Silent Installation

To perform a silent installation:

1. Create a response file. See "Creating a Response File" for instructions.
2. Download and unzip the WebRTC Session Controller software on the machine on which you will run the silent installation. See "General Pre-Installation Tasks" for instructions.
3. Copy the response file you created to the machine on which you will run the silent installation.
4. On the machine on which you will run the silent installation, run the following command:

   ```
   java -jar wsc_generic.jar -silent -responseFile fullpathtoresponsefile
   ```

   Where *fullpathtoresponsefile* is the full path including the filename of your response file

   For example:

   ```
   java -jar wsc_generic.jar -silent -responseFile /home/user/responsefile.txt
   ```

The WebRTC Session Controller Installer checks for all required software and writes errors to a log file if it detects any missing or unavailable components, or if there are any connectivity-related issues.

See "Troubleshooting a Signaling Engine Installation" for information about WebRTC Session Controller Signaling Engine installer logs.

# Next Steps

After you install Signaling Engine, you must configure a Signaling Engine domain. See "Creating and Configuring a WebRTC Session Controller Signaling Engine Domain" for instructions.

> **Note:** To uninstall Signaling Engine, you run the command *Oracle_home*/**oui/bin/desinstall.sh** and click **Next** and then click **Deinstall**. See "Oracle Universal Installer Deinstallation Screens" in *Oracle Fusion Middleware Installing with the Oracle Universal Installer* for more information.
>
> You will have to remove the *Oracle_home* directory before you can reinstall the software.

# 6

# Creating and Configuring a WebRTC Session Controller Signaling Engine Domain

This chapter describes the steps required to create your WebRTC Session Controller Signaling Engine (Signaling Engine) domain after your software has been successfully installed.

## About Domains and Domain Configuration

After you install the Signaling Engine software, you must create a domain for your deployment. Before continuing in this chapter, you need to understand WebLogic domains and clustering, and the domain topologies available for use with WebRTC Session Controller Signaling Engine

- To learn about WebLogic domains and clustering, see "WebLogic Server Domains" and "WebLogic Server Clustering" in *Understanding Oracle WebLogic Server*.
- To learn about the domain topologies available for use with WebRTC Session Controller Signaling Engine, see "Understanding Signaling Engine Installation Topologies".

## Configuring Your Signaling Engine Domain

This section provides instructions for creating a WebRTC Session Controller domain using the configuration wizard. For more information on other methods available for domain creation, see "Additional Tools for Creating, Extending, and Managing WebLogic Domains" in *Creating Domains Using the Configuration Wizard*.

The following topics are covered in this section:

- About Signaling Engine Domain Types
- Starting the Configuration Wizard
- Configuring a WebRTC Session Controller Domain

### About Signaling Engine Domain Types

There are a selection of domain templates to choose from but only two are relevant to Signaling Engine:

- **Oracle Communications WebRTC Session Controller Replicated Domain**

  The Replicated Domain template enables you to create a replicated WebRTC Session Controller Signaling Engine domain. The Replicated Domain topology is

designed for use with WebRTC applications that require high levels of scalability, availability, and performance.

- **Oracle Communications WebRTC Session Controller Basic Domain**

  The Basic Domain template enables you to create a simple Signaling Engine domain. Such a domain configuration can be used during development where it is more convenient to deploy and test applications on a single server.

In addition, each domain type can be extended to add Diameter support. For more information, see the discussion of WebRTC Session Controller Diameter Rx to Policy Charging and Rules (PCRF) configuration in the *Oracle Communications WebRTC Session Controller System Administrator's Guide*.

## Recommendations and Requirements for Replicated Domains

If you are configuring a Signaling Engine replicated domain, keep the following tips in mind for ease of deployment:

- Keep the directory structures the same on each machine in the cluster to simplify deployment.
- Install the Signaling Engine binaries on each machine in the cluster.
- Install any required patches manually on each machine in the cluster; they are not distributed automatically.
- Synchronize the file *Domain_home*/**security**/SerializedSystemIni.dat between each managed server residing on separate machines before starting the managed servers.
- Create the replicated domain on your admin server, and copy that domain to each of your cluster machines.

---

**Note:** If the administration server is already started before you copy its domain to the cluster servers, delete the existing domains on each cluster server:

```
rm -rf Domain_home/servers/
```

---

## Starting the Configuration Wizard

To begin domain configuration, navigate to the *Oracle_home***/wlserver/common/bin** directory and start the Fusion Middleware Configuration Wizard:

```
./config.sh
```

---

**Note:** If, while installing Signaling Engine using the GUI installation wizard, you checked the Automatically Launch the Configuration Wizard check box, the Domain Configuration wizard will already be running.

---

## Configuring a WebRTC Session Controller Domain

Follow the instructions in this section to configure the domain using the Configuration Wizard.

1. On the Configuration Type screen, select **Create a New Domain**.

   In the Domain Location field, specify your Domain home directory.

It is recommended that you locate your Domain outside the Oracle home directory. This directory structure will help you avoid issues when you need to upgrade or re-install your software.

> **Tip:** More information about the Domain home directory can be found in "Choosing a Domain Home" in *Planning an Installation of Oracle Fusion Middleware*.
>
> More information about the other options on this screen can be found in "Configuration Type" in *Creating Domains Using the Configuration Wizard*.

2. Click **Next**.

   The Templates window appears.

3. On the Templates screen select one of the following templates:

   - **Oracle Communications WebRTC Session Controller Replicated Domain**
   - **Oracle Communications WebRTC Session Controller Basic Domain**

     When you select **Oracle Communications WebRTC Session Controller Basic Domain**, the Configuration wizard selects the **WebLogic Coherence Cluster Extension** template automatically.

     ---

     **Note:** The **Basic WebLogic Server Domain** template is selected by default and cannot be deselected.

     More information about the options on this screen can be found in "Templates" in *Creating Domains Using the Configuration Wizard*.

     ---

4. Click **Next**.

   The Administrator Account screen appears.

   On the Administrator Account screen, specify the user name and password for the default WebLogic Administrator account for the domain. This account is used to connect to the domain's Administration Server.

   > **Tip:** You must make a note of the user name and password you choose to enter here; you will need this in order to be able to start and access the Administration Server.

5. Click **Next**.

   The Domain Mode and JDK screen appears.

   On the Domain Mode and JDK screen:

   - Select **Development** or **Production** in the Domain Mode field.
   - Select **Oracle Hotspot JDK** in the JDK field or choose a different supported JDK.

     See "Signaling Engine Software Requirements" for information on supported JDKs.

   Selecting **Production Mode** on this screen gives your environment a higher degree of security, requiring a user name and password to deploy applications and to start the Administration Server.

**Tip:** In production mode, a boot identity file can be created to bypass the need to provide a user name and password when starting the Administration Server. For more information, see "Creating a Boot Identity File for an Administration Server" in *Administering Server Startup and Shutdown for Oracle WebLogic Server*.

6. Click **Next**.

   There are several advanced options you can choose to configure on the Advanced Configuration screen:

   - **Administration Server**

     Checking this option lets you configure the listen address of the Administration Server.

   - **Node Manager**

     Checking this option lets you configure Node Manager. For more information on Node Manager, see "Node Manager Overview" in *Oracle Fusion Middleware Administering Node Manager for Oracle WebLogic Server*.

   - **Managed Servers, Clusters and Coherence**

     Checking this option lets you configure the Managed Servers, Clusters, and also lets you configure the machine and assign Managed Servers to the machine.

     **Tip:** If you want to configure dynamic clusters, see the following:

     - "Overview of Dynamic Clusters" in *Understanding Oracle WebLogic Server*.
     - "Creating Dynamic Clusters" in *Administering Clusters for Oracle WebLogic Server*.

   - **Deployments and Services**

     Checking this option lets you customize how application deployments and services are targeted to servers and clusters.

   Check the advanced options you wish to configure.

   ---

   **Note:** If you are configuring a Oracle Communications WebRTC Session Controller Replicated Domain in **Production Mode**, you must select the following advanced options:

   - **Administration Server**
   - **Managed Servers, Clusters and Coherence**

   If those options are not configured the WebLogic server will not start.

   ---

7. If you have not checked any advance options, click **Next** and continue to step 8. Otherwise, click **Next** and follow this sub procedure:

   a. If you have chosen the Administration Server advanced option, the Administration Server screen appears.

      For more information on the options available on this screen, click the **Help** button and refer to the Wizard's online help. You can also refer to

"Administration Server" in *Oracle Fusion Middleware Creating WebLogic Domains Using the Configuration Wizard*.

---

**Note:** If you are configuring a **Oracle Communications WebRTC Session Controller Replicated Domain** in **Production Mode**, you must enter the **Listen Address** for the Administration Server.

---

Make any updates required and click **Next**.

**b.** If you have chosen the Node Manager advanced option, the Node Manager screen appears.

For more information on the options available on this screen, click the **Help** button and refer to the Wizard's online help. You can also refer to "Node Manager" in *Oracle Fusion Middleware Creating WebLogic Domains Using the Configuration Wizard*.

Make any updates required and click **Next**.

**c.** If you have chosen the Managed Servers, Clusters and Coherence advanced option, the Managed Server screen appears. Otherwise, skip to step 8.

For more information on the options available on this screen, click the **Help** button and refer to the Wizard's online help. You can also refer to "Managed Servers" in *Oracle Fusion Middleware Creating WebLogic Domains Using the Configuration Wizard*.

---

**Note:** If you are configuring a **Oracle Communications WebRTC Session Controller Replicated Domain** in **Production Mode**, you must select or enter the **Listen Address** for each engine in the Managed Servers screen.

---

Make any updates required and click **Next**.

**d.** The Clusters screen appears.

For more information on the options available on the Clusters screens, click the **Help** button and refer to the online help. You can also refer to "Clusters" in *Oracle Fusion Middleware Creating WebLogic Domains Using the Configuration Wizard*.

---

**Note:** You must also enter the **Cluster Address** for the engine cluster.

---

Make any updates required and click **Next**.

**e.** The Assign Servers to Clusters screen appears.

For more information on the options available on the Assign Servers to Clusters screen, click the **Help** button and refer to the online help. You can also refer to "Assign Servers to Clusters" in *Oracle Fusion Middleware Creating WebLogic Domains Using the Configuration Wizard*.

---

**Note:** You must also enter the **Cluster Address** for the engine cluster.

---

Make any updates required and click **Next**.

f. If you have included Coherence in the WebRTC Session Controller Installation, the Coherence Clusters screen appears. If you have not included Coherence, this screen will be skipped.

   For more information on the options available on the Coherence Clusters screens, click the **Help** button and refer to the online help. You can also refer to "Coherence Clusters" in *Oracle Fusion Middleware Creating WebLogic Domains Using the Configuration Wizard*.

   Make any updates required and click **Next**.

g. The Machines screen appears.

   For more information on the options available on the Machines screen, click the **Help** button and refer to the online help. You can also refer to "Machines" in *Oracle Fusion Middleware Creating WebLogic Domains Using the Configuration Wizard*.

   Make any updates required and click **Next**.

h. The Assign Servers to Machines screen appears.

   For more information on the options available on the Machines screen, click the **Help** button and refer to the online help. You can also refer to "Assign Servers to Machines" in *Oracle Fusion Middleware Creating WebLogic Domains Using the Configuration Wizard*.

   Make any updates required and click **Next**.

8. The Configuration Summary screen appears.

   The Configuration Summary screen contains the detailed configuration information for the domain you are about to create. Review the details of each item on the screen and verify that the information is correct.

   You can go back to any previous screen if you need to make any changes, either by using the **Back** button or by selecting the screen in the navigation pane.

   Domain creation will not begin until you click **Create**.

9. The Configuration Success screen will show the following items about the domain you just configured:

   - Domain Location
   - Administration Server URL

   You must make a note of both items as you will need them to start the servers and access the Administration Server.

   Click **Finish** to close the configuration wizard.

## Starting the Signaling Engine Servers

After configuration is complete, in order to access the tools with which you can manage your domain, you must start the necessary servers. See the following topics for more information:

- Starting the Node Manager
- Starting the Administration Server
- Starting the Managed Servers

## Starting the Node Manager

To start your per-domain Node Manager, go to the *Domain_home*/bin directory.

Start Node Manager as shown below, using nohup and nm.out as an example output file:

```
nohup ./startNodeManager.sh > nm.out&
```

**Note:** It is recommended that you install Node Manager to run as a startup service. This allows Node Manager to start up automatically each time the system is restarted.

For more information, see "Running Node Manager as a Startup Service" in *Administering Node Manager for Oracle WebLogic Server*.

## Starting the Administration Server

To start the Administration Server, go the *Domain_home*/bin directory and run:

```
./startWebLogic.sh
```

If you selected **Production Mode** on the Domain Mode and JDK screen in step 5, you will be prompted for the login credentials of the Administrator user as provided on the Administrator Account screen in step 4.

**Tip:** For more information about starting the Administration Server, see "Starting and Stopping Administration Servers" in *Administering Oracle Fusion Middleware*.

In production mode, a boot identity file can be created to bypass the need to provide a user name and password when starting the Administration Server. For more information, see "Creating a Boot Identity File for an Administration Server" in *Administering Server Startup and Shutdown for Oracle WebLogic Server*.

## Starting the Managed Servers

**Note:** Before starting WebRTC Session Controller Managed Servers, copy *domain_home*/**security**/**SerializedSystemIni.dat** file from the Admin server to same directory on the managed server.

To start the Managed Servers, go the *Domain_home*/bin directory and run the following command:

```
./startManagedWebLogic.sh managed_server_name admin_server_url
```

Replace *managed_server_name* with the name of the Managed Server you want to start.

Replace *admin_server_url* with the full URL of the Administration Server, as provided on the Configuration Success screen in step 9.

Below are sample commands used to start wsc-se_server_1 and wse-se_server_2 on UNIX operating systems:

```
./startManagedWebLogic.sh wsc-se_server_1 t3://host.example.com:7001 &
./startManagedWebLogic.sh wsc-se_server_2 t3://host.example.com:7001 &
```

**Tip:** For more information about starting Managed Servers, see "Starting and Stopping Managed Servers" in *Administering Oracle Fusion Middleware*.

### Example: Starting a Replicated Domain Configuration

The following example shows how to start a Signaling Engine replicated domain. For this example, the deployment model consists of three machines:

- An Admin server with the IP address 10.1.1.1
- A clustered machine with the IP address 10.1.1.2, hosting an engine, **engine1**.
- A second clustered machine with the IP address 10.1.1.3, hosting an engine, **engine2**.

To start the example replicated domain:

1. On the Admin server, 10.1.1.1 execute

```
cd Domain_home/bin
./startWeblogic.sh
```

2. On the first clustered machine, execute:

```
cd Domain_home/bin
./startManagedWebogic.sh engine1 t3://10.1.1.1:7001
```

3. On the second clustered machine, execute:

```
cd Domain_home/bin
./startManagedWebogic.sh engine2 t3://10.1.1.1:7001
```

## Next Steps

After you have configured the Signaling Engine domain, you must complete Signaling Engine post-installation tasks. See "WebRTC Session Controller Signaling Engine Post-Installation Tasks" for instructions.

# 7

# WebRTC Session Controller Signaling Engine Post-Installation Tasks

This chapter provides instructions for Oracle Communications WebRTC Session Controller Signaling Engine (Signaling Engine) post-installation tasks.

Before continuing, you must complete the tasks in the following chapters:

- Installing WebRTC Session Controller Signaling Engine
- Creating and Configuring a WebRTC Session Controller Signaling Engine Domain

## Overview of Signaling Engine Post-Installation Tasks

After installing Signaling Engine, you must complete the following post-installation tasks:

- Enable DNS Server Lookup
- Set the SIP Proxy Server and Registrar IP Address
- Configure WebRTC Session Controller Authentication
- Configure the Coherence Security Framework

Before continuing, you must start your Signaling Engine servers. See "Starting the Signaling Engine Servers" for more information.

## Enable DNS Server Lookup

You must enable DNS server lookup on your Signaling Engine installation to ensure that the **maddr** parameter in SIP headers is processed properly.

To enable DNS server lookup:

1. Start your Signaling Engine servers if they are not already running. See "Starting the Signaling Engine Servers" for more information.
2. Navigate to the WebLogic Server administration console and log in with your administrator username and password:

   `http://hostname:port/console`

   **Note:** The default administration console port is 7001.

3. In the Domain Structure pane, select **SipServer**.

4. Select the Configuration tab and then select the General sub tab.
5. In the General sub tab, scroll down and select the **Enable DNS Server Lookup** option.
6. Click **Save**.
7. Log out of the administration interface.

## Set the SIP Proxy Server and Registrar IP Address

To set the SIP proxy server and registrar IP address:

1. Start your Signaling Engine servers if they are not already running. See "Starting the Signaling Engine Servers" for more information.
2. Navigate to the WebRTC Session Controller Signaling Engine console and log in with your administrator username and password:

   `http://hostname:port/wsc-console`

   **Note:** The default Signaling Engine console port is 7001.

3. From the **Home** tab, select **Signaling Engine**.
4. Click **Edit**.
5. In the **Proxy Registrar URI** field, enter the URI of the proxy registrar, as **sip:***hostname***:***port*.
6. Click **Save** to save your changes.

## Configure WebRTC Session Controller Authentication

You must configure WebRTC Session Controller to provide authentication for WebRTC clients, even if you only want to allow unauthenticated access. For more information, see "Configuring WebRTC Session Controller Authentication" in *Oracle Communications WebRTC Session Controller System Administrator's Guide*.

## Configure SSL Hostname Verification

If your installation is configured to use Secure Socket Layer (SSL) support, you need to set the hostname verification to **None** using the WebLogic console.

To configure SSL hostname verification:

1. Start your Signaling Engine administration server if it is not already running.
2. Navigate to the WebLogic Server Administration Console and log in with your administrator user name and password:

   `http://hostname:port/console`

   where *hostname* is the name of your WebRTC Session Controller server and *port* is the Administration Console access port.

   **Note:** The default Administration Console port is 7001.

3. In the Domain Structure pane, expand **Environment**, and select **Servers**.
4. In the Summary of Servers pane, select the **Configuration** tab.
5. For each of your servers in the Servers table, do the following:
   a. Click the server name in the table, for example, **AdminServer**.
   b. In the Settings for *ServerName* pane, select the **SSL** tab.
   c. Expand the **Advanced** settings at the bottom of the pane.
   d. From the **Hostname Verification** list box entries, select **None**.
   e. Click **Save**.

No restart is required after making this change.

## Configure the Coherence Security Framework

If you have created a clustered domain, you must enable the Coherence Security Framework. For instructions, see "Enabling the Oracle Coherence Security Framework" in *Securing Oracle Coherence*.

## Next Steps

You must install WebRTC Session Controller Media Engine servers. See "WebRTC Session Controller Media Engine Installation Overview" for instructions.

# 8

# Upgrading WebRTC Session Controller Signaling Engine from an Earlier Version

This chapter provides instructions for upgrading to Oracle Communications WebRTC Session Controller Signaling Engine (Signaling Engine) from an earlier version.

## About Upgrading Signaling Engine

Because of architectural changes in the underlying WebLogic platform, there is no automated upgrade path from earlier versions of Signaling Engine to version 7.2. Instead, you must install the latest version of Signaling Engine on each of your servers and manually synchronize the following configurations from old to new:

- **SSL Configuration**: If you are using SSL, configure SSL on every node machine using the WebLogic configuration console.
- **Security Providers**: Configure your security providers using the WebLogic configuration console. For more information see "Configuring WebRTC Session Controller Authentication" in the *Oracle Communications WebRTC Session Controller Administrator's Guide*.
- **Signaling Engine Applications**: Using the Signaling Engine console, configure your applications in the Applications tab to match those from your old installation. For more information see the *Oracle Communications WebRTC Extension Developer's Guide*.
- **Signaling Engine Packages**: Using the Signaling Engine console, configure your packages in the Packages tab to match those from your old installation. For more information see the *Oracle Communications WebRTC Extension Developer's Guide*.
- **Groovy Script Library**: If you have customized your Signaling Engine Groovy Script Library, migrate your changes to the Script Library tab in the Signaling Engine console.
- **Proxy Registrar Addresses**: Using the Signaling Engine console, set the proxy registrar address in the Script Library tab. For more information see the *Oracle Communications WebRTC Extension Developer's Guide*.
- **Media Engine Nodes**: Using the Signaling Engine console, configure information for your Media Engine nodes. For more information see the *Oracle Communications WebRTC Extension Developer's Guide*.

## Changes in the Latest Version of Signaling Engine

This section lists the major changes affecting upgrade deployments in the latest version of Signaling Engine.

- By default, the Signaling Engine (and the Media Engine) communication is based on HTTPS protocol.
- The SIP container is JSR 359 compliant; and the Java Development Kit version is 1.8.
- The **register** package now supports **hibernate** and **iceEnquiry** for sessions.
- Signaling engine configuration is now grouped by integration, runtime, resource limit and log level parameters.
- WebRTC Session Controller now supports multitenancy profile configuration; and Signaling engine Groovy script library supports more groovy properties.

  For more information on multitenancy, see "About Multitenancy" in *WebRTC Session Controller System Administrator's Guide*.

# High Availability

For information on high availability environments, see "Failure Prevention and Automatic Recovery Features" in the *Oracle Communications WebRTC Session Controller System Administrator's Guide*.

# An Example Replicated Domain Upgrade

The following example shows the theoretical steps involved in upgrading a replicated domain. For this example, the deployment model consists of three machines:

- An Admin server, with the IP address 10.1.1.1.
- A clustered machine with the IP address 10.1.1.2, hosting an engine, **engine1**.
- A second clustered machine with the IP address 10.1.1.3, hosting an engine, **engine2**.

To upgrade the replicated domain:

1. Install Signaling Engine on each machine in your current 7.1 topology. (This example case uses three machines.)

   Follow the instructions in "Installing WebRTC Session Controller Signaling Engine".
2. You can set up a new domain in your 7.2 installation in one of two ways:
   - Create a new domain. See "Creating a New WebRTC Session Controller 7.2 Domain".
   - Migrate an existing 7.1 domain. See "Migrating an Existing WebRTC Session Controller 7.2 Domain".
3. Configure the 7.2 WebRTC Session Controller. See "Configuring WebRTC Session Controller Administration Console".
4. Verify the updated configurations. See "Verifying the Updated Configurations".
5. Test the managed servers. See "Testing the Updated Configuration".

## Creating a New WebRTC Session Controller 7.2 Domain

Create **Oracle Communications WebRTC Session Controller Replicated Domain** domains on each machine using the same configuration parameters from your old Signaling Engine installation.

See "Creating and Configuring a WebRTC Session Controller Signaling Engine Domain".

**Note:** You must select the **Administration Server** and **Managed Servers, Clusters and Coherence** advanced options when configuring the replicated domain otherwise Signaling Engine will fail to start.

**Note:** Replicas are no longer required. You can either re-purpose **replica1** and **replica2** as additional engines or decommission them completely.

## Migrating an Existing WebRTC Session Controller 7.2 Domain

Migrate an existing WebRTC Session Controller 7.1 domain as described in this section. Complete the steps for every server.

**Pack the existing WebRTC Session Controller 7.1 domain for every server**

1. Go to *WSC71_home***/wlserver/common/bin** directory, where *WSC71_home* is the directory where you installed WebRTC Session Controller 7.1.
2. To package the domain, run the **pack.sh** command. In the replicated domain example, this command is:

```
./pack.sh -domain=/wsc71_home/user_projects/domains/replica_domain
-template="/wsc71_home/user_projects/domains/packedreplicadomain.jar"
-template_name="pack wsc7.1 replica_domain"
```

**Unpack the 7.1 domain data in the 7.2 installation area for every server**

1. Go to *WSC72_home***/wlserver/common/bin** directory, where *WSC72_home* is the directory where you installed WebRTC Session Controller 7.2.
2. To unpack the domain information, run the **unpack.sh** command. In the replicated domain example, this command is:

```
./unpack.sh -domain=/wsc72_home/user_projects/domains/replica_domain
-template="/wsc71_home/user_projects/domains/packedreplicadomain.jar"
```

**Reconfigure the 7.2 domain in your 7.2 Installation area for every server**

1. Go to *WSC72_home***/wlserver/common/bin** directory, where *WSC72_home* is the directory where you installed WebRTC Session Controller 7.2.
2. To reconfigure the domain information, run the **reconfig.sh** command. In the replicated domain example, this command is:

```
./reconfig.sh -domain=/wsc71_home/wlserver /wsc72_home/user_
projects/domains/replica_domain
```

The **Fusion Middleware Reconfiguration Wizard** is displayed.

**Complete Fusion Middleware Reconfiguration Wizard steps for every server**

1. In the **Fusion Middleware Reconfiguration Wizard**, select this domain to upgrade. Click **Next**.
2. The **Reconfiguration Setup Progress** window displays the progress of the setup. The display ends with the URL to the location for the domain:

```
Completed!
```

```
Core WLS Indfrastructure Reconfigured Successfully.
Domain Location:/.../user_projects/domains/replica_domain
```

Click **Next**.

3. In the **Domain Mode and JDK** window, verify the selections.

   Click **Next**.

4. In the **Advanced Configuration** window, check the managed servers and deployments, if necessary.

   Click **Next**.

5. In the **Configuration Summary** window, check the entries.

   If necessary click the **Back** command button and correct the entries. When you are done, click **Reconfig**.

6. The **Reconfiguration Progress** window displays the progress of the setup. The display ends with the following statements:

   ```
   Performing Post Domain Creation tasks...
   Domain Reconfiguration Applied Successfully.
   ```

   Click **Next**.

7. The **Reconfiguration Success** window lists the URLs for the Domain and the Admin Server.

> **Tip:** Note down the paths to this domain and to the Admin Console.

## Configuring WebRTC Session Controller Administration Console

The administration console of WebRTC Session Controller in this (7.2) release is more robust and enables you to configure multitenancy. You can update the configuration in the WebRTC Session Controller in one of two ways:

- Manually configuring the Settings. See "Updating the Configuration Settings Manually".
- Migrating the configurations using utilities:
  - Converting to the 7.2 Configuration with a Migration Utility
  - Extracting the Groovy Code from the 7.1 Configuration

### Updating the Configuration Settings Manually

If you are manually configuring the settings for the 7.2 Signaling Engine in the new console using the settings in the 7.1 domain, refer to the descriptions of the fields in "Configuring WebRTC Session Controller" in the *WebRTC System Administration Guide*.

### Converting to the 7.2 Configuration with a Migration Utility

The **migration.tar.gz** compressed TAR file provided with this release contains the **migrate.sh** utility. This utility generates a configuration file called **wsc-config-migrated.xml** by migrating the **wsc-config.xml** configuration file to be compatible with the 7.2 version. After running this utility, you can use the **wsc-config-migrated.xml** file in your 7.2 installation.

To generate the new **wsc-config-migrated.xml** configuration file:

**Extract the migrate.sh utility**

1. Go to the directory where you downloaded the **migration.tar.gz** file.
2. Right-click on **migration.tar.gz** file and extract its contents.

   The current directory contains the **migrate.sh** file.

**Migrate the 7.1 wsc-config.xml file**

1. If you are not already where the **migrate.sh** file is located, go to that directory.
2. To generate the updated configuration file, enter this command:

   **sh migrate.sh** *<path_to_your_current_wsc-config.xml>*

   Where, *path_to_your_current_wsc-config.xml* is the pathname to the location of the current **wsc-config.xml** file.

   For example,

   ```
   sh migrate.sh ../server/AdminServer/wsc/wsc-config.xml
   ```

   When this command completes, the current directory contains a configuration file called **wsc-config-migrated.xml**. This file can be used in the 7.2 installation.

## Extracting the Groovy Code from the 7.1 Configuration

The **groovyExtractor.tar.gz** compressed TAR file provided with this release contains the **extractGroovy.sh** utility. This utility generates a Groovy file called **extractedGroovy.groovy** by extracting all the Groovy code from the 7.1 version of **wsc-config.xml** file.

To extract the groovy code from the 7.1 version of the **wsc-config.xml** file:

**Extract the extractGroovy.sh utility**

1. Go to the directory where you downloaded the **groovyExtractor.tar.gz** file.
2. Right-click on **groovyExtractor.tar.gz** file and extract the contents of this TAR file.

   The current directory contains the **extractGroovy.sh** utility.

**Extract the Groovy Code**

1. Go to the directory where the **extractGroovy.sh** utility is located.
2. To generate the Groovy output file containing the groovy code from the 7.1 installation, enter this command:

   **sh extractGroovy.sh** *<path_to_your_current_wsc-config.xml>*

   Where, *path_to_your_current_wsc-config.xml* is the pathname to the location of the current **wsc-config.xml** file.

   For example,

   ```
   sh extractGroovy.sh ../server/AdminServer/wsc/wsc-config.xml
   ```

   When this command completes, the current directory contains a Groovy file called **extractedGroovy.groovy**. This file contains the groovy code from the 7.1 domain.

## Verify and Configure the Settings in the 7.2 Signaling Engine

Complete these steps in the WebRTC Session Controller 7.2 Administration Console:

1. Login to the Signaling Engine WebLogic console. For the example case, **http://10.1.1.1:7001/wsc-console.**
2. Click **Edit**
3. If not already selected, click the **Home** Tab.

   Access each of its sub tabs:

   - **Signaling Engine**
   - **Media Engine**

   Verify that the values are configured appropriately with respect to your current 7.1 configuration.
4. Click the **Application Profiles** tab.

   The console displays a list of the current application names and three sub-tabs, **Profile**, **Packages**, and **Library**.

   Select every application name listed under **Name** and do the following:

   a. Review the contents of the **Profile** tab for that application name entry.

   b. Click **Packages**. The associated Groovy scripts are displayed. Click **Validate** and correct all errors. Click **Save**.

   c. Click **Library**. The Groovy script library is displayed. Click **Validate** and correct all errors. Click **Save**.
5. Click the **Packages** tab.

   The console displays three panes, **Packages** which lists the supported packages, **Criteria** which lists every message criteria configured for a selected package, and a pane to display the Groovy script for a selected message criteria belonging to that package.

   Select every package name listed under **Packages** and for each package, do the following:

   a. Select a criteria in the **Criteria section** tab for the selected package name.

   b. Review the Groovy script displayed for the selected criteria for this package. Click **Validate** and correct all errors.

   c. Click **Save**.

   Repeat the 3 sub steps for every criteria associated with the selected package.

   For the descriptions of the fields see "Configuring WebRTC Session Controller" in the *WebRTC System Administration Guide*.

## Verifying the Updated Configurations

To verify the updated configurations in the administration console for WebRTC Session Controller 7.2:

**Start the WebLogic 7.2 Admin Server**

1. Stop the old Admin server.
2. Go to the domain home for WebLogic Session Controller 7.2.

   Start the new WebLogic Administration Console using the command:

   ```
   cd Domain_home/bin
   ./startWeblogic.sh
   ```

If you are using SSL, configure SSL on every node.

**Verify the SSL Configuration**

1. Login to the Signaling Engine WebLogic console. For the example case, **http://10.1.1.1:7001/console.** Do the following:
   a. Expand the **Environment** node in the Domain Structure panel and click **Servers**.
   b. In the **Configuration** tab of the **Summary of Servers** pane, select an engine server, check **SSL Listen Port Enabled**, and that the **SSL Listen Port** is correct.
   c. Click **Save** to save your configuration.
   d. Repeat for the remaining engine servers.

**Verify the Configuration for the Security Providers**

1. Login to the Signaling Engine WebLogic console. For the example case, **http://10.1.1.1:7001/console**.

   Verify that the WebLogic security realm use the same parameters from your old installation.

   For more information on configuring a security realm, see "Configuring WebRTC Session Controller Authentication" in *Oracle Communications WebRTC Session Controller System Administrator's Guide*.

**Configure Coherence Security Based on Your Deployment**

1. Log in to the WebLogic Administration Console for the 7.2 release.
2. In the **Domain Structure**, select **Coherence Clusters**.

   The **Summary of Coherence Clusters** pane is displayed to the right.
3. In the **Coherence Clusters** table, select **defaultCoherenceCluster**.
4. From the tabs displayed under **Settings for defaultCoherenceCluster**, select **Security**.
5. Configure Coherence security access each of the following tabs and provide input, as required:
   a. **General**
   b. **Services**
   c. **Cache**

**Verify the Application Router**

1. Log in to the WebLogic Administration Console for the 7.2 release.
2. In the **Domain Structure**, select **Sip Server**.
3. In the **SIP Server** pane, select **Configuration**.
4. Under the **Configuration** tab, select **Application Router.**
5. Verify the Application Router(AR) features.

## Testing the Updated Configuration

Verify that all of the updates are in effect by doing the following:

1. Stop the signaling engine and replica servers from your old installation.

2. In the new installation, start all the managed servers and verify the original deployment work as expected.
3. Configure a third-party load balancer as required.

   **Note:** Signaling Engine no longer includes load balancer support.

4. Verify that your applications work as expected.

# 9

# WebRTC Session Controller Media Engine Installation Overview

This chapter provides an overview of the Oracle WebRTC Session Controller Media Engine (ME) software's supported hardware, licensing, and system management.

**Note:** If you are installing a patch set for the Media Engine, refer to the *Release Notes* and the *ReadMe* files that accompany the patch set for information on installing it.

## Supported WebRTC Session Controller Media Engine Third-Party Devices

This section lists the supported ME third-party servers.

The following platforms have been certified for use with the ME:

- Sun Netra X5-2
- Sun Server X5-2
- Sun Netra X3-2
- HP DL160 G9
- NN2610
- NN2620

The following VM platforms have been certified for use with the ME:

- OVM 3.3.1
- VMware ESXi 5.5
- Xen 3.4.3
- KVM on OL7

## Information on Media Engine Software and Licensing

As part of each download, Oracle provides the USB Creation Utility with the ME software.

You must provide a USB stick with 4GB storage to handle Oracle software downloads. Oracle has tested a variety of USB sticks available from current suppliers and manufacturers. Most USB sticks manufactured today will work.

For complete information on accessing the Oracle download server, creating an installation USB stick, and commissioning ME systems, refer to "Creating and Commissioning USB Sticks".

### Obtaining Your License

If you are not using royalty-bearing codecs, you should be using the ME's default shipping license.

The default license enables the maximum number of sessions for the system. In the past, the software would stop allowing new sessions once your licensed maximum was reached (i.e., 100). The system software no longer relies on the license to apply an upper limit. This can be problematic because, depending on the server hardware you are using, the system may not be capable of supporting a higher number of sessions. You may want to edit your config file and add the parameters for the maximum number of media sessions to ensure that you do not exceed the capabilities of your hardware. This applies to most deployments running a small number of sessions on smaller third-party hardware that could potentially have a problem if traffic increased to a number larger than the system is able to handle.

The following are royalty-bearing codecs supported by the ME:

- AMRWB
- AMRNB
- G723
- G729

### License Expirations and Renewals

If your customer-specific license comes with an expiration date, the ME system generates an event when the license nears the expiration date. Contact your Oracle sales representative to complete the purchase of the features that you are testing. These expiring licenses should apply only to users testing royalty-bearing codecs for transcoding.

## System Management

Before you install the system, you should decide on the management tool(s) that you want to use to configure and monitor the system. This will help you decide where you need to create connections based on your equipment and network resources.

System management capabilities include the following secure management interfaces:

- The ME command line interface (CLI) from a local console, Telnet, or SSH connection
- The ME Management System, a graphical user interface (GUI) that supports remote management using a Web browser
- Simple Network Management Protocol (SNMP) using third party SNMP MIB compiler/browser applications

For information on configuring the management options, refer to the *Oracle Communications WebRTC Session Controller Media Engine Object Reference*.

# Installing the Media Engine

Starting with release 7.2, the ME runs on Oracle Linux and uses yum to install and update RPM files. In prior releases, the media engine came with its own custom kernel.

> **Note:** While the ME operates under Oracle Linux, it is not certified to operate under other Linux environments.

You must have Oracle Linux Release 7.0 or higher installed on your hardware prior to installing the ME.

To install the ME you must:

- Install Oracle Linux version 7.0 or higher
- Download and copy the ME file to a USB stick
- Mount the ME software onto your hardware
- Configure a yum repository on which to point Oracle Linux
- Install the ME appliance

## Installing Oracle Linux 7

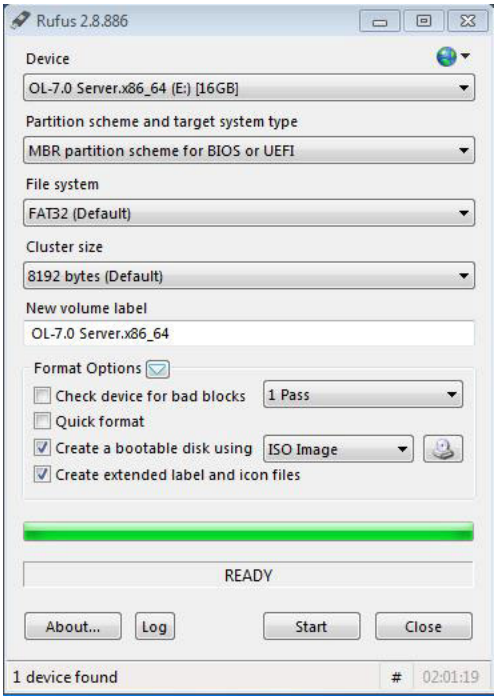
Before you can install the ME, you must have Oracle Linux installed on your hardware. You can either install Oracle Linux via a USB stick or a DVD. This guide documents installing Oracle Linux via a USB.

> **Note:** When you install the ME, you must have Oracle Linux installed on your hardware. You can either install Oracle Linux via a USB stick or a DVD. This guide documents installing Oracle Linux via a USB.

For a much more comprehensive and thorough description of installing Oracle Linux 7, see *https://docs.oracle.com/cd/E52668_01/E54695/E54695.pdf*.

To install Oracle Linux via a USB stick:

1. Download “Oracle Linux 7.X for x86 64 bit ISO image” from http://edelivery.oracle.com/linux
2. Create a bootable USB stick that contains the full Oracle Linux 7 ISO image. The following example uses Rufus 2.8 software to create the bootable USB stick.

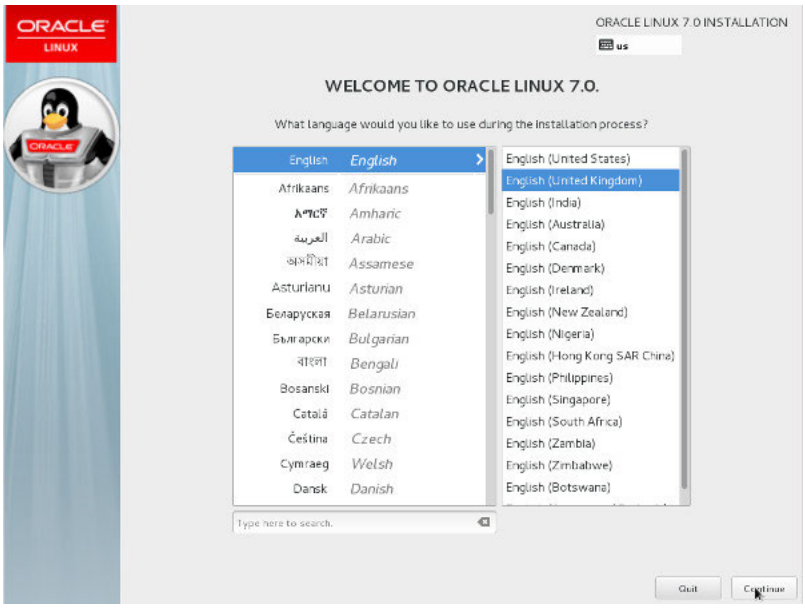



**Note:** You can also install Oracle Linux via a DVD by downloading "Oracle Linux 7.1 for x86 64 bit ISO image" from http://edelivery.oracle.com/linux and burning the *.iso image onto a DVD.

3. Insert your bootable Oracle Linux 7 USB drive onto your hardware.
4. Boot the system from the boot image by selecting Boot from usb from the boot menu options.

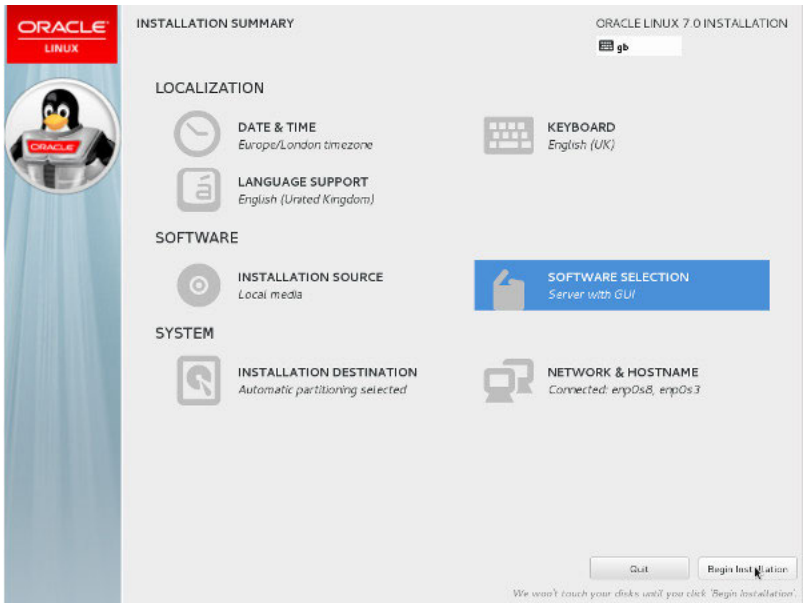

5. Select **Install Oracle Linux 7.0** and hit <*Enter*>.

6. Select the appropriate language and select **Set keyboard to default layout for selected language**. Click **Continue**.




The "Installation Summary" screen appears.

7. Complete any marked items. Depending on your requirements, you may also need to alter the default settings by clicking on the relevant links.




8. Click **Installation Destination**.
9. Select the local disks you want to use for the installation.


**Note:** The installation program does not make any changes to any of the disks and storage until you click **Begin Installation** on the "Installation Summary" screen.

- Choose the disks on which you want to install Oracle Linux from the "Local Standard Disks" section. A tick icon displays next to the selected disks.
- Ensure the **Automatically configure partitioning** option is selected (by default, this option is selected).
- At the bottom of the screen, the system displays how much disk space you need for the software you have selected. With automatic partitioning, you may not have sufficient space to install the software if the disk is already partitioned. If you need to free some disk space, select **I would like to make additional space available** and click **Done**.

**Note:** You must have disk size of at least 50 GB.

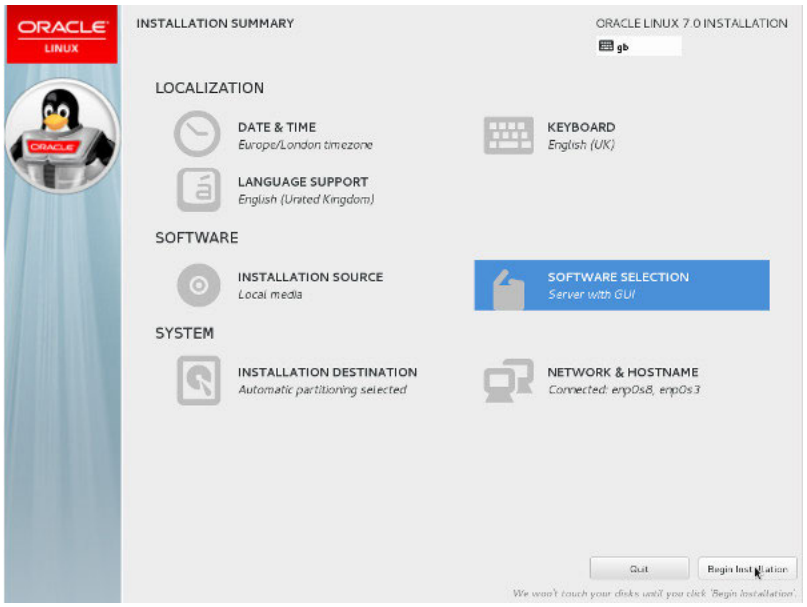
The "Reclaim Disk Space" window appears.


**Note:** If there is still insufficient disk space when you click Done, the system prompts you to free disk space.

- Once you have selected the disks you want to use, click **Delete all** to free disk space and then click **Reclaim Space**.

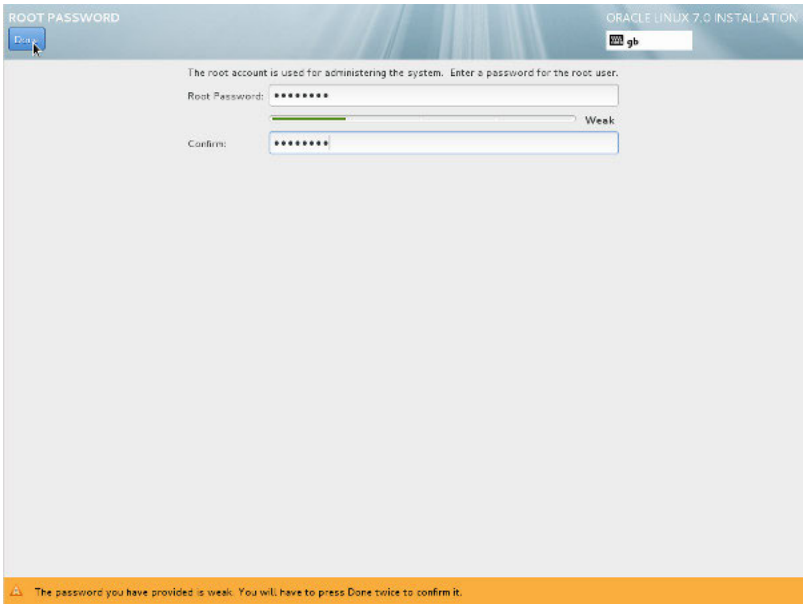
  For more information on configuring partitioning, see https://docs.oracle.com/cd/E52668_01/E54695/E54695.pdf.

10. Click **Begin Installation** once you have completed any necessary updates to the default configuration.



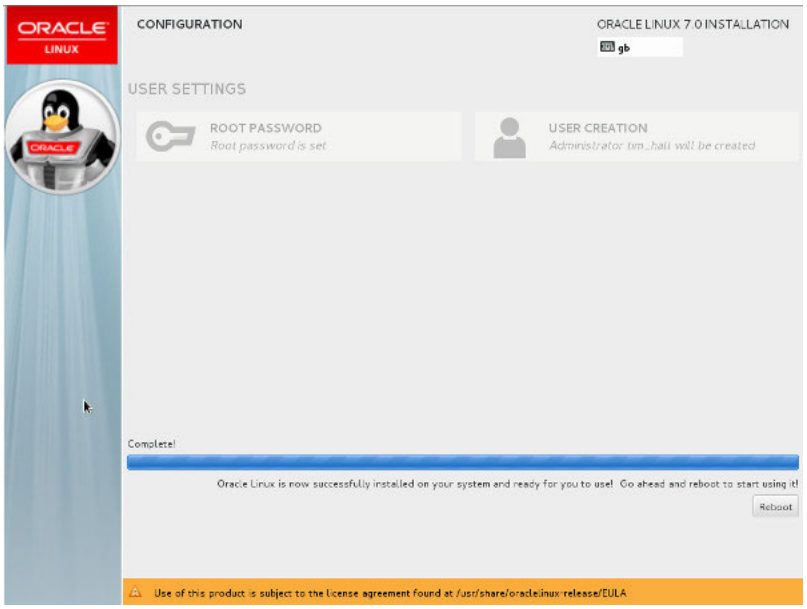

11. Click **Root Password**.


**12.** Enter the root password and click **Done**.




Linux installs.

**13.** Click **Reboot** when prompted.




14. Run the following command to rename the Ethernet devices according to your system BIOS once you successfully complete the installation.

    ```
    grubby --args=net.ifnames=0 --update-kernel=ALL
    ```

15. Reboot the system.

## Obtaining the Media Engine Installation File

Before you can install the ME, you must first download the ISO files you need and copy them to a USB stick.

Software can be downloaded from either the Oracle Software Delivery Cloud or the My Oracle Support Patches and Updates tab.

**To access the Oracle Software Deliver Cloud:**

1. Access the https://edelivery.oracle.com link.
2. Select the **Sign In/Register** tab and enter your **username** and **password**.

> **Note:** If you are a new user, you must create an account.

3. Click the checkbox to agree to the to the **Oracle Trial License Agreement and Export Restrictions** and click **Continue**.
4. Select the Oracle Communications product pack.
5. Select the Acme Packet OS platform and click **Go**.
6. Download the following file and click **Download**.
   - Oracle Communications WebRTC Session Controller 7.2 Installation Repository
7. Copy the file onto a USB stick.

**To access the Oracle Support Software Patches and Updates:**

1. Log into the My Oracle Support Portal.
2. Select the **Patches and Updates** tab.

3. Select the **Search** tab and click **Product or Family (Advanced)**.
4. **Product:** Enter **Oracle Communications Application Session Controller**.
5. **Release:** Enter **WebRTC Session Controller 7.2 Installation Repository**.
6. Click **Search**. The available distribution formats appear and include the following information:
   - Patch Name
   - Description
   - Release
   - Platform (Language)
   - Classification
   - Product
   - Prerequisite Requirement
   - Size
   - Download Access
7. Select the distribution format that you require.
8. Click either **Download** to download the file or **Read Me** to view the Build Notes for this patch.
9. Copy the file onto a USB stick.

### Mounting the Media Engine File

Once you have the installation file, you must install it on your hardware.

To mount the ME installation file:

1. Insert the USB stick onto your hardware and locate the USB stick partition to mount (for example, /dev/sdd1).

```
[root@localhost]# sudo <fdisk -l>
```

2. Create a mount point and mount the installation file.

```
[root@localhost]# sudo mkdir /mnt/usb
[root@localhost]# sudo mount /dev/sdd1 /mnt/usb
```

3. Extract the files via "unzip".

**Note:** The following is an example and uses example values only.

```
[root@localhost]#cd /mnt/usb
[root@localhost]# unzip 370m4p0-2016-03-24_22-51-16
Archive: 370m4p0-2016-03-24_22-51-16.zip
inflating:
370m4p0/kernel-uekcov-debuginfo-3.8.13-118.4.1.370m4p0.69387.el7uekcov.x86_
64.rpm
inflating: 370m4p0/apache-commons-fileupload-1.3.1-4.el7.noarch.rpm
inflating: 370m4p0/crtmpserver-690-3.el7.i686.rpm
inflating: 370m4p0/slotmap-2.2.3-1.el7.i686.rpm
inflating: 370m4p0/jre-8u65-fcs.1.el7.i686.rpm
inflating: 370m4p0/asc-app-emblcrimport-E3.7.0.M4P0-69407.el7.i686.rpm
inflating: 370m4p0/apache-commons-fileupload-javadoc-1.3.1-4.el7.noarch.rpm
inflating: 370m4p0/portlet-2.0-api-1.0-9.el7.noarch.rpm
```

```
inflating:
370m4p0/kernel-uekcov-debuginfo-common-3.8.13-118.4.1.370m4p0.69387.el7uekcov.x
86_64.rpm
inflating: 370m4p0/libipp-7.0-2.el7.i686.rpm
inflating:
370m4p0/kernel-uekcov-covmodule-2.8.1-3.8.13.118.4.1.370m4p0.68883.el7uekcov.x8
6_64.rpm
inflating: 370m4p0/postgresql-odbc-09.03.0100-2.el7.cov1.i686.rpm
inflating: 370m4p0/asc-appliance-E3.7.0.M4P0-69407.el7.i686.rpm
inflating: 370m4p0/kernel-uekcov-3.8.13-118.4.1.370m4p0.69387.el7uekcov.x86_
64.rpm
inflating: 370m4p0/libunwind-1.1-5.el7.2.i686.rpm
inflating:
370m4p0/kernel-uekcov-firmware-3.8.13-118.4.1.370m4p0.69387.el7uekcov.x86_
64.rpm
inflating: 370m4p0/asc-selinux-policy-E3.7.0.M4P0-29.el7.noarch.rpm
inflating: 370m4p0/asc-rescue-stick-E3.7.0.M4P0-29.el7.i686.rpm
inflating: 370m4p0/asc-app-samples-E3.7.0.M4P0-69407.el7.i686.rpm
inflating: 370m4p0/portlet-2.0-api-javadoc-1.0-9.el7.noarch.rpm
inflating: 370m4p0/h264bitstream-0.1.6-1.el7.i686.rpm
inflating: 370m4p0/rtmpdump-2.3-1.el7.i686.rpm
inflating: 370m4p0/repodata/filelists.xml.gz
inflating: 370m4p0/repodata/primary.xml.gz
inflating: 370m4p0/repodata/repomd.xml
inflating: 370m4p0/repodata/other.xml.gz
```

**Note:** If your system does not have the "unzip" package installed, execute the **yum install unzip** command.

## Configuring a Yum Repository

Oracle Linux 7 uses "yum" to install and update RPM files. Yum uses a URL to point at repositories. The URL must be pointed to the mounted USB directory you create when you mount the installation file, described in <blue>Mounting the Media Engine File.

To configure a yum repository:

1. Create the repository entry using root permissions.

```
cat >/etc/yum.repos.d/asc.repo <<'!'
[asc-base]
name=Application Session Controller Base Repository
baseurl=file:///mnt/usb/370m4p0
gpgcheck=0
enabled=1
```

### Configuring an Unconnected Network to a Yum Repository

If your system is not connected to a network, you must point the baseurl in the "public-yum-ol7.repo" file to an Oracle Linux 7 ISO DVD image partition.

To configure an unconnected network to a yum repository:

1. Point the baseurl in the "public-yum-ol7.repo" file to an Oracle Linux 7 ISO DVD image partition.

**Note:** The following is an example and uses example values only.

```
vi /etc/yum.repos.d/ public-yum-ol7.repo
[ol7_latest]
name=Oracle Linux $releasever Latest ($basearch)
baseurl=http://servername/$basearch/
gpgkey=file:///etc/pki/rpm-gpg/RPM-GPG-KEY-oracle
gpgcheck=1
enabled=0
[ol7_u2_base]
name=Oracle Linux $releasever Update 2 installation media copy ($basearch)
baseurl=file:///mnt/cdrom/Packages
gpgkey=file:///etc/pki/rpm-gpg/RPM-GPG-KEY-oracle
gpgcheck=1
enabled=1
[ol7_UEKR3]
name=Latest Unbreakable Enterprise Kernel Release 3 for Oracle Linux
$releasever ($basearch)
baseurl=http://servername/$basearch/
gpgkey=file:///etc/pki/rpm-gpg/RPM-GPG-KEY-oracle
gpgcheck=1
enabled=0
```

## Installing the Media Engine Appliance

Once the yum repository is configured, you can install the ME appliance onto an Oracle Linux 7 system.

**To install the ME appliance:**

1. Enter the **yum install** command with root permissions.

   *[root@localhost]# yum install asc-appliance*
2. Reboot your Oracle Linux system so that it starts with the new ME appliance kernel required for the ME to operate properly.
3. Once you've successfully completed the installation, unmount the ME installation USB stick.

   ```
   [root@localhost]# umount /dev/sdd1 /mnt/usb/
   ```

# 10

# Quick Commissioning New Media Engine Systems

This chapter provides the basic information that allows you to configure Media Engine (ME) software after you have physically installed the system in your network. Commissioning enables an ME system or compatible third-party device to process WebRTC and WebRTC-Session Initiation Protocol (SIP) sessions.

## Prerequisites to Quick Commissioning

Before using the information in this chapter, make sure that you have properly installed and cabled the system. The following ME documents provide additional information on configuring ME services, as well as how manage the system using the ME CLI and the ME Management System.

- *Oracle Communications WebRTC Session Controller System Administrator's Guide*
- *Oracle Communications WebRTC Session Controller Media Engine Object Reference*

Additionally, the *Oracle Communications WebRTC Session Controller Release Notes* provides important information about the software that you should review before commissioning a system in your network.

Steps 1 through 5 cover the tasks and services for getting the system up and running on an IP network so that the Ethernet interfaces can process WebRTC and WebRTC-SIP sessions. When enabled on an IP network, you can manage the system and its configuration remotely over the Internet using the ME Management System.

Steps 6 through 10 cover the tasks that allow you to control and monitor WebRTC and WebRTC-SIP sessions, as well as store call detail records and recordings.

## Building the Configuration File

The ME configuration file (**cxc.cfg**) is made up of configuration objects and property settings that control how the system processes and manages WebRTC and WebRTC-SIP traffic. As you open these objects and set properties using the CLI or the ME Management System, the software builds a configuration hierarchy of objects that are applied to WebRTC and WebRTC-SIP sessions. You can display this configuration hierarchy using the **show** and **show -v** (verbose) commands.

For new users, as well as for users who are adding functionality to their configuration, you will need to open configuration objects using the **config** command to enable the default settings for those objects, *even if you choose not to edit any of their associated properties*. For example, if you need to enable the **ICMP** protocol and its default settings, you simply open the object and execute **return**, as shown in the session below.

Notice that the ICMP object has been added to the configuration hierarchy at the end of the session on the eth4 interface.

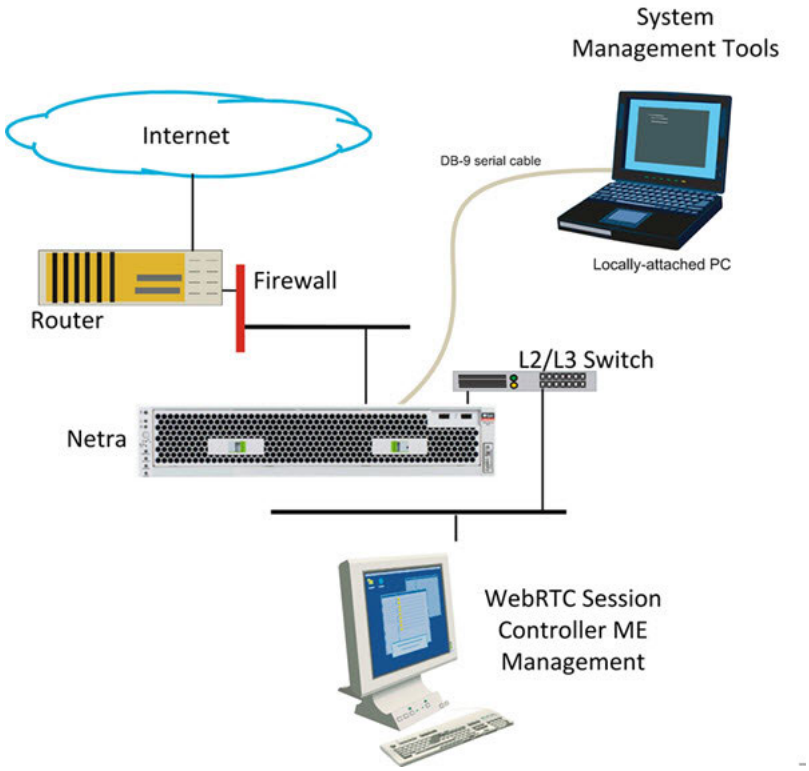
```
config> config box interface eth4
config interface eth4> config ip 172.26.2.14
config ip 172.26.2.14> config icmp
config ip 172.26.2.14> return
config interface eth4> return
config box> return
config> show -v
interface eth4
  admin enabled
  mtu 1500
  arp enabled
  speed 1Gb
  duplex full
  autoneg enabled
  ip 172.26.2.14
   admin enabled
   ip-address dhcp
   geolocation 0
   metric 1
   classification-tag
   security-domain
   address-scope
   filter-intf disabled
   icmp
    admin enabled
    limit 10 5
```

To remove an object from the configuration hierarchy, use the CLI or ME Management System **delete** command. For more information on the **delete** command, see the *Oracle Communications WebRTC Session Controller Media Engine Object Reference*.

## Basic Network Topology

Figure 10–1 illustrates a network topology using the ME with a directly-attached PC for initial setup, and the ME Management System for remote access using a graphical user interface.

*Figure 10–1 Media Engine Network Topology*




# Step 1. Configuring Basic IP Connectivity

Before you can manage an ME system remotely over the Internet using the ME Management System or over a Telnet or SSH connection, you need to locally assign an IP address to one of the Ethernet interfaces, **eth0**, **eth1, eth2,** or **eth3.** If you are setting up the device remotely, you will also need to configure an IP route, a route to a destination host or network, and a gateway IP address.

If you are using the ME Management System, you will also need to know the assigned IP address on one of the Ethernet ports to manage the ME configuration. The ME Management System application runs directly on the ME system over the Internet.

Example 10–1, "Configuring Basic IP Connectivity" shows a CLI session creates and enables an IP interface named **192.168.124.5,** sets the static IP address and network mask, configures an IP route (if connecting remotely), and enables Web access on this IP interface. You will need to enable ICMP on the ME IP interface before you can use the **ping** command from your console to test the device as a responding node on the network. Use the **show -v** command to display the configuration.

## CLI Session

***Example 10–1 Configuring Basic IP Connectivity***

```
NNOS-E> config box
config box> set hostname local2610
config box> config interface eth1
config interface eth1> config ip mgmt-int
Creating 'mgmt-int'
```

```
config mgmt-int> set admin enabled
config mgmt-int> set ip-address static 192.168.124.5/24
config mgmt-int> config routing
config routing> config route internetGateway
Creating 'route internetGateway'
config route internetGateway> set destination default
config route internetGateway> set gateway 192.168.124.3
config route internetGateway> return
config routing> return
config ip mgmt-int> config web
config web> set admin enabled
config web> set port 80
config web> return
config mgmt-int> config icmp
config icmp> set admin enabled
config icmp> top
config> save
config> show -v
```

## Using the Setup Script

An optional configuration setup script called *cxc.setup* is now included with newly shipped systems. After installing a new system, you can run the script directly from the NNOS-E> prompt, as shown in Example 10–2.

***Example 10–2 Using the Setup Script***

```
CLI Session
NNOS-E> config setup
set box\hostname: <name>
config box\interface: eth1
set box\interface eth1\ip a\ip-address: <ipAddress/mask>
config box\interface eth1\ip a\ssh (y or n)? n
config box\interface eth1\ip a\web (y or n)? y
config box\interface eth1\ip a\routing\route: <routeName>
set box\interface eth1\ip a\routing\route localGateway\gateway:
<ipAddress>
set box\cli\prompt: <newPrompt>
Do you want to commit this setup script (y or n) y
Do you want to update the startup configuration (y or n)? y
```

The script presents a set of questions to help you with the initial system configuration. The information in the script includes the following:

- Local hostname
- IP interface names and addresses
- SSH and Web access
- Default route and any additional static routes per interface for remote management
- User-defined CLI prompt

Every ME system has a minimum of two Ethernet interfaces. Any Ethernet interface on the system can be used for management traffic, however, Oracle recommends the use of eth1, as eth0 is reserved for fault-tolerant clustering with other ME systems. Management traffic is also supported on any interface that is carrying private or public network traffic. This means that it would be possible to use eth1 to carry WebRTC and WebRTC-SIP traffic and management traffic.

**Note:** The /cxc directory on the ME system may include vendor-specific scripts that address unique startup configuration requirements. Specify the name of the script on the command line following the config setup command. For example: NNOS-E> config setup vendor.setup Check the /cxc directory for any vendor-specific setup files included with your system.

## Enabling Network Access

To ensure you can manage the system using services such as Telnet or the ME Management System, you must configure the ME system so that it is available on the network. You need to create a default (or static) IP route, a route to a destination host or network, and a gateway IP address.

After you configure the static route, enable ICMP and then use the **ping** command at the top-level of the CLI to test network accessibility.

For more information configuring static routes and enabling ICMP, see "Using the Setup Script".

## Defining a Default Route and Gateway IP

If you are setting up the box remotely, you must configure an IP route, a route to a destination host or network, and a gateway IP address.

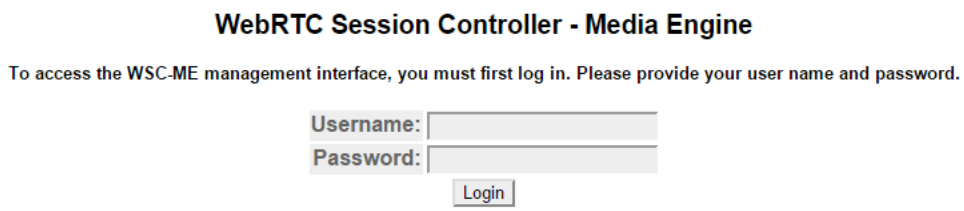
See "Step 1. Configuring Basic IP Connectivity" for the example CLI session that shows the routing context and the route named *internetGateway*. This is the default route that uses 192.168.124.3 as the default gateway.

## Launching the Media Engine Management System

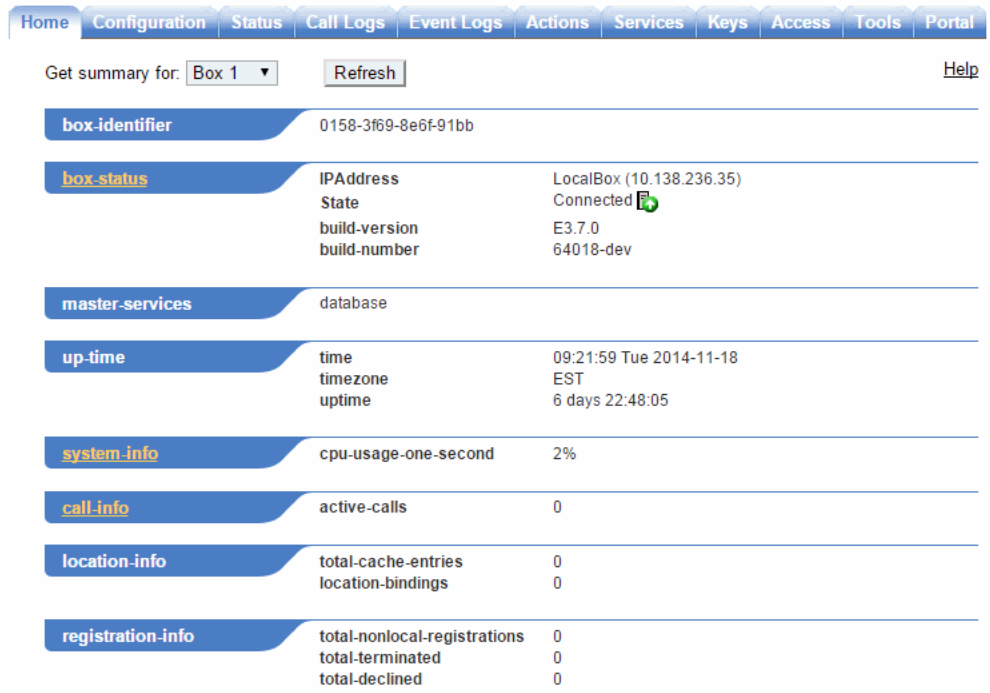
In addition to the CLI, you can use the ME Management System to configure the ME. To access the ME using the ME Management System, open an HTTP or secure HTTP window (HTTPS) to the IP address of the Eth0 port on the ME system. For example:

```
https://192.168.124.5
```

You should see the Oracle ME Log In window, illustrated in the following image.




By default, there are no user accounts configured on a new system. This means any value can be entered in for username & password, or leave the fields blank and click **Login**. Once you log in, the ME Management System main page appears.




The remaining steps in this chapter use the ME Management System to commission the ME.

## Changing the Linux Root Password

To change the Linux root password, use the **secret root** action. When prompted, specify and confirm the new password. For example:

```
NNOS-E>secret root
password:*******
confirm:*******
Success!
NNOS-E>
```

**Note:** The password must be at least four characters long.

For more information on the **secret root** action, see the *Oracle Communications WebRTC Session Controller Media Engine Objects and Properties Reference Guide*.

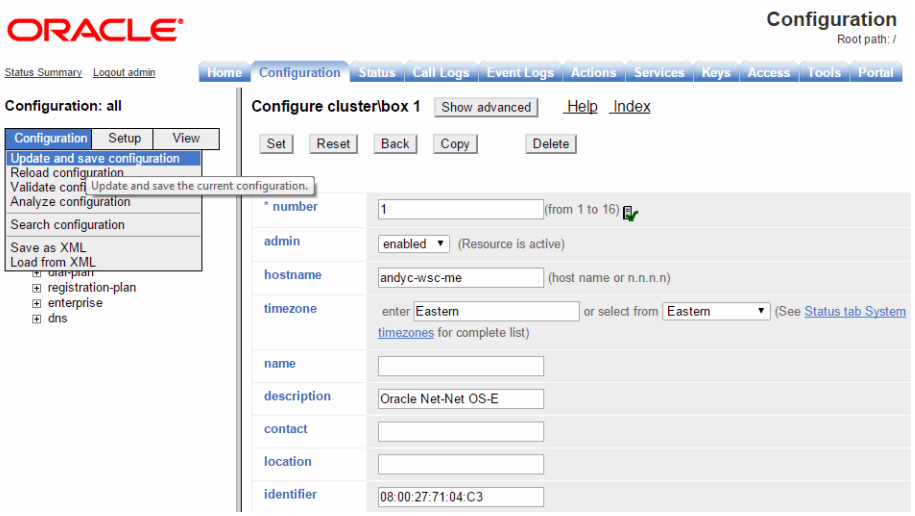
# Step 2. Configuring Advanced IP Connectivity

Use the **Configuration** tab or the CLI to configure several additional Ethernet interfaces, as covered in "Step 1. Configuring Basic IP Connectivity". As a security device, the ME uses a default setting of **disabled** for these objects in the configuration file. This means that you must enable each interface. These objects include:

- **SSH**: To enable SSH client connectivity on the interface
- **Media ports**: To enable a range of port numbers for on the interface
- **SIP**: To enable SIP traffic on the interface)

When editing Ethernet interface and examining each object using the ME Management System, note that many of the objects are already visible, but they are not yet enabled. For these objects to actually be enabled on the ME system, you must select the object and save the configuration.

After editing an interface configuration, elect **Set**, then **Update & save configuration**, as illustrated in the following image.




When you select **Configuration/Update and save configuration** you will be asked "Do you want to update the live configuration?" followed by "Do you also want to save the live configuration?" Click **OK** for both questions to ensure that the configuration is properly saved to the ME configuration file, *cxc.cfg*.

The following steps are necessary to set some specific parameters for the objects listed above:

1. Select the Configuration **Cluster/ Box 1/Interface Eth0/IP local** object on the left menu tree. Under the **General** field, edit the Media Ports properties as desired, then click **Set**.
2. Under the **Other Properties** field, edit the SSH properties. Accept the defaults by clicking **Set**.
3. Select **SIP** from the menu tree. Enter the following values for each fields:
   - admin: enabled (default)
   - NAT translation: disabled (default)
   - UDP port: Select **Add UDP port**, accept the defaults, then click **Finish->Set**.
   - TCP port: Select **Add TCP port**, accept the defaults, then click **Finish->Set**.
   - TLS port: Select **Add TLS port**, accept the defaults, then click **Finish->Set**.
   - Certificate: blank (default)

When you are finished editing the SIP fields, select **Set->Configuration/Update and save configuration**.

## Step 3. Creating User Accounts for Basic Access

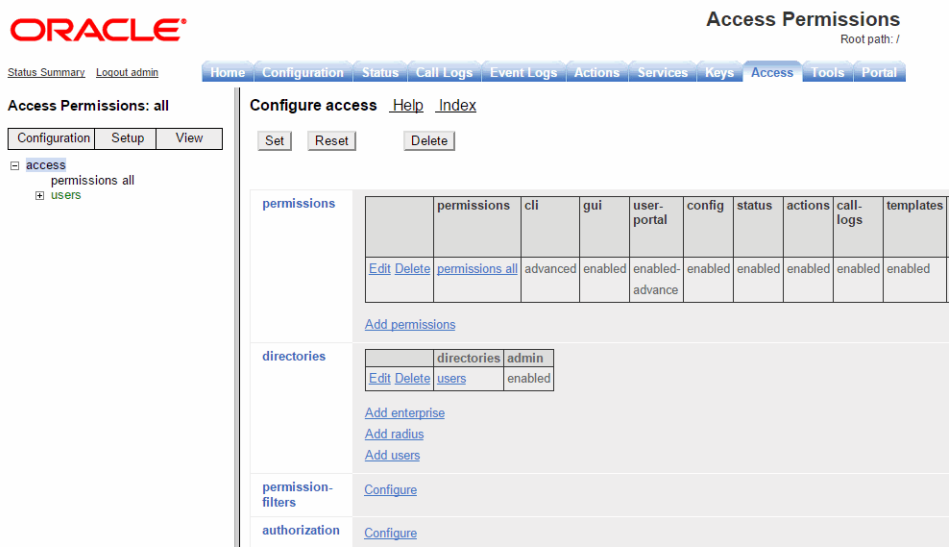
By default, the ME does not contain any predefined user accounts. This means it is possible to access the management interfaces without entering any login credentials

(username and password). To properly secure ME/SE integration, however, you must configure two users. First configure a management user with access to all ME functionality. Then configure a web services user with access to web-services, actions, and statuses only.

> **Note:** You must configure the management user first. If you configure the web services user first, you will not be able to access the CLI or web UI.

If you want to create a user account at this time, follow the steps below. If not, go directly to Step 4.

1. Using the ME Management System, select the **Access** tab, then select **Access** from the left menu pane. The Access Permissions/Configure Access page appears.




2. Under **permissions**, select **Add permissions** and create a permissions group called *super-user* and accept all default settings with all permission types enabled.

   Select **Set**, then select **Update and save configuration** from the Configuration pull-down in the left pane**.**

3. From the **Directories** object, select **Add users**. Accept the default setting of enabled.

4. Select **Add user** and enter the required **name** and **password** of your choice, then re-enter the password to **confirm** your original password entry. In the **permissions** field, choose the permissions group that you just created (super-user).

5. Click **Create.** Select **Configuration->Update and save configuration**.

These steps created a username and password for a super-user account. Future attempts to log in to the ME (using the CLI or the ME Management System) will require that you specify these login credentials. If needed, you can also create user accounts with one or more of the super-user permissions.

## Step 4. Enabling Master Services

The **master-services** configuration enables directory, accounting, database and registration services to run on the system. Perform the following steps to configure these master services:

1. Select the **Services** tab, then select **master-services** from the left menu pane.
2. Accept the default settings for **cluster-master**, **directory, accounting**, **database** (with **Show advanced** button selected), and **registration**. Click **Set**.

After you have configured all five services, select Configuration->**Update and save configuration**. The completed Master Services configuration appears.

## Step 5. Configuring Basic Services

The **Services** configuration enables event logging and virus scanning services to run on the ME. Perform the following steps to configure event logging on the system.

1. Select the **Services** tab then select Services from the left menu pane.
2. On the Configure services page, select **event-log** from the menu pane, accept the defaults and click **Set**. Under the **event-log** configuration, additional options are available that you can configure.

   You can direct the event logs to one or more of the following locations:

   - A syslog server
   - An ASCII file in an ME directory
   - A database on the ME system
   - An external database
3. In the **file** object, click **Edit**, then enter the name *event-log* in the text block. Click **Set**.

   This configures event logging so that messages are written to the local file named *event-log*.

## Step 6. Enabling the Virtual System Partition (VSP)

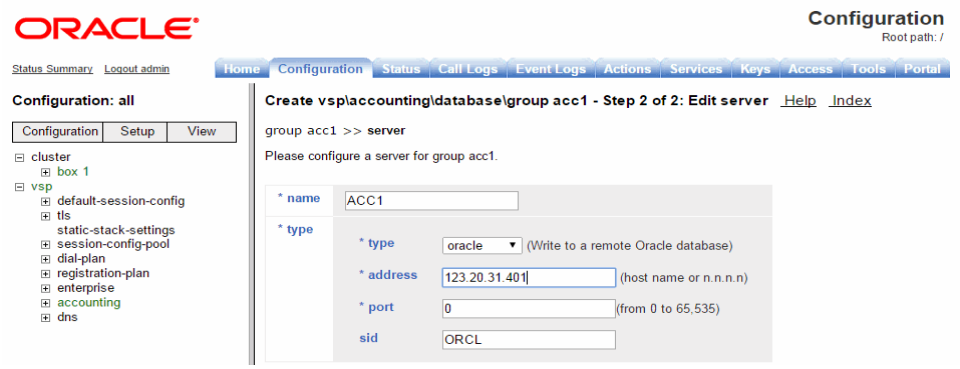
The ME virtual system partition (VSP) is the part of the system that holds the comprehensive customer-defined configuration that controls how the system processes, stores, directs, and routes WebRTC and WebRTC-SIP traffic. The VSP is where you can create session configurations, registration and dial plans, and policies that handle session message traffic that the system will receive and forward to a call destination, authentication and accounting database, service provider or enterprise server, and so on.

Using the ME Management System, perform the following steps.

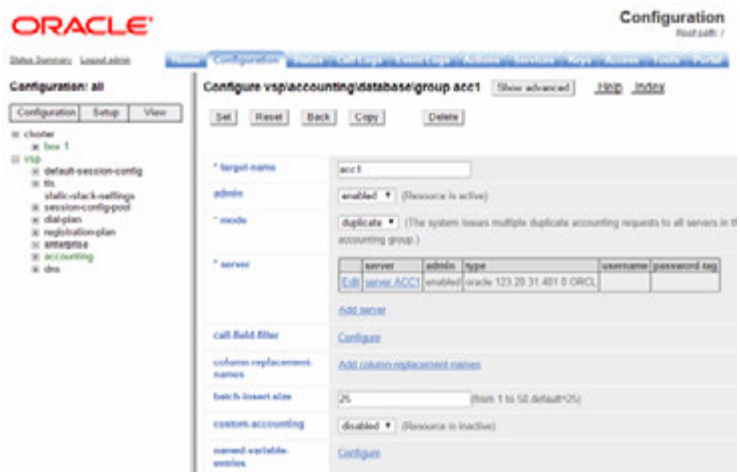
1. Select the **Configuration** tab, then select **vsp** from the menu to open the Configure vsp page.
2. Under the general heading:, change the **admin** state to **enabled**.
3. Click **Set,** then select **Configuration->Update and save configuration**.

# Step 7. Configuring the Accounting Environments

1. Select the ME Management System **Configuration** tab, then select **vsp->accounting** from the menu to display the Configure vsp\accounting page.
2. Under **targets**, click **Configure** next to the **database** field and set the **admin** property to enabled.
3. Click **Add group** and enter the **target-name** and **mode** property. Click **Next**. The Edit server screen appears.




4. Enter the database's name in the **target-name** field and select **Create** to display the Configure database group page.




5. Click **Edit** beside the **server** field and configure the following settings:
   - **admin**: enabled
   - **name:** localdb
   - **type:** Select **local**
   - **username**: postgres
   - **password-tag:** postgres

---

**Note:** If you set the server type to local, using the local database as the accounting target, set the username and the password-tag to postgres. If you edit the username and password-tag properties to anything other than postgres, data will not be written to the database.

---

For information about password tags, refer to the *Oracle Communications WebRTC Session Controller Media Engine Object Reference*.

6. Click **Set,** then select **Configuration->Update and save configuration**.

## Step 8. Configuring the Media Engine to Process SIP Traffic

The next step is to configure a default system policy that allows the ME to process SIP traffic. By default, and for security purposes, the ME does not allow any SIP traffic to pass.

1. Select the Configuration tab, then select **vsp->default-session-config** from the menu to display the vsp/default-session-config page.
2. In the **sip-directive** object, change the directive policy to **allow**, if not already set. This allows SIP traffic to traverse the ME system. Click **Set**.
3. Scroll down to the **media** object. Change the **anchor** property to **enabled.** Accept all other default settings.
4. Click **Set**, then select **Configuration->Update and save configuration.**

## Step 9. Reviewing the Configuration

Once you have completed Steps 1 through 8, review the configuration to make sure it is accurate. A quick way to do this is to scan the ME Management System navigation tree to make sure there is an entry for each of the objects that you configured.

The following image is a listing of the Configuration and Services objects configured as part of basic ME commissioning. If you are using the CLI, run the **show -v** command from the ME prompt to display the configuration that you just created. The following image displays the configuration and services navigation trees.


## Generating a Certificate

The ME communicates with the SE via HTTPS by default. While you can use a default self-signed certificate, you may need to generate one. This section describes generating a certificate.

### Creating a Self-Signed Certificate and Key Pair from the Media Engine

Use the ME software to generate a cryptographic key pair and a self-signed X.509 certificate in PKCS#12 format. Once you create a self-signed certificate, you can

generate the Certification Signing Request, a portion of which is required by the CA upon submission of their form.

Under the **Actions** tab, select **cert-gen** from the commands list. The Generate new key and certificates page appears.

Complete the fields on the Generate new key and certificates page.

- **keyFile**: Specify the name and directory path of the resulting key name that you want to use, along with the p12 or .pfx file extension.

  Example: /cxc/certs/<*myNetworkKey*>.p12

- **passphrase**: Specify a password to be associated with the self-signed certificate. The text that you specify is encrypted in the certificate.

- **alias**: Specify the FQDN of the ME system using this certificate, such as abc123.example.com. Omit HTTP:// and HTTPS://. This allows the certificate to be referenced.

  **Note:** The value (FQDN) you enter for the **alias** field must be identical to the value you enter for the **common-name** field.

- **common-name**: Specify the FQDN of the ME system using this certificate, such as abc123.example.com. Omit HTTP:// and HTTPS://. Do not use your personal name in this field. The common name is a component attribute of the certificate's distinguished name.

- **days-valid**: Enter the number of days for which the certificate is valid. If your certificate if effective for one year, then enter the number 365.

- **country**: Select the ISO country code: US (United States), AU (Australia), IN (India), IT (Italy), UK (United Kingdom), CA (Canada). The country is a component attribute of the certificate's *distinguished name*.

- **alternate-name**: Typically a name that complies with the ASN.1 specification, such as a DNS name, IP address, or URI.

- **organization**: Enter the name under which your business is legally registered. The listed organization must be the legal registrant of the domain name in the certificate request. If you are enrolling as a small business/sole proprietor, enter the certificate requestor's name in this field, and the DBA (doing business as) name in the **organizational-unit** field. The organization is a component attribute of the certificate's *distinguished name*.

- **organizational-unit**: Use this field to differentiate between divisions within an organization. For example, "Engineering" or "Human Resources." If applicable, you may enter the DBA (doing business as) name in this field. The organizational unit is a component attribute of the certificate's *distinguished name*.

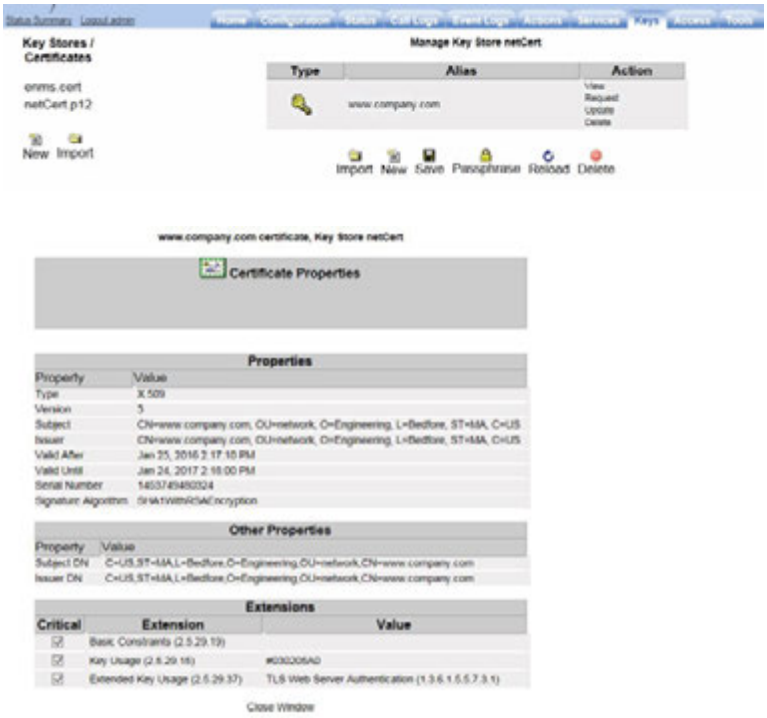
- **state**: If in the US, enter one of the fifty state names, in full, where your organization is located, such as Massachusetts; if outside the US, enter the full name of a province or region.

- **locality**: Enter the name of a city.

When you are finished filling out the fields, click **Invoke**. A Success message appears.

### Viewing the Certificate

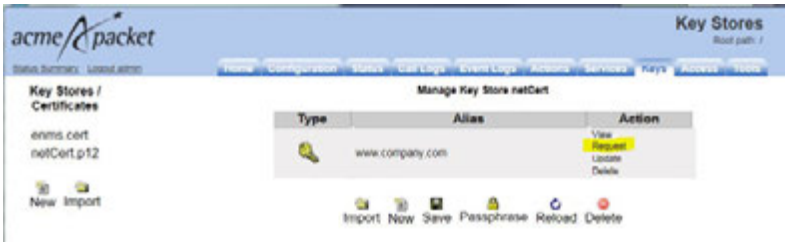
To view the self-signed certificate, select the **Keys** tab from the main menu bar, then select the keyFile that you just created from the Key Stores list on the left. Click **View** to display the Certificate Properties page.

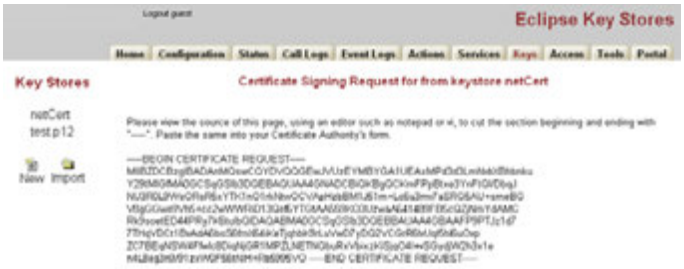



## Generating a Certification Signing Request

After you have created the self-signed certificate from the previous step, you must generate a certification signing request (CSR) that you can submit to the CA for the X.509 certificate. Select the **request** action.

The Generate Certificate Signing Request page and the resulting certificate signing request appear. Enter the password that you created in the previous step in the **passphrase** field and click **Generate Certificate Signing Request**. Select the key store file on the left and enter your password when prompted. Manage Key Store is displayed; click on **Request** under **Action**.






Click the **Export to File** button to save the CSR provided by the CA to external file.

If you choose to create a CSR in a PEM-formatted file, select the **cert-request** action.

Follow the instructions on the Certificate Signing Request page to copy and paste the text into the certificate application form provided by the CA.

Complete the fields on the Generate Certification Request page, using the same settings that you invoked from Step 1, as follows:

- **keyfile**: Specify the name and ME directory path of the resulting key name that you want to use, along with the p12 or .pfx file extension. Example: /cxc/certs/<*myNetworkKey*>.p12
- **passphrase:** Specify a password to be associated with the certificate issued by the CA. The text that you specify will be encrypted in the CSR.
- **alias:** Enter an alias.

> **Note:** The value you enter for this field must be identical to the value you enter for the **common-name** field.

- **csr-file:** Specify the name and directory path of the resulting CSR file. This is the file from which you will cut and paste the required information for the CA at the time that you submit the certificate request. By default, the CSR file resides in the directory named /cxc/certs.

When you are finished filling out the fields, click **Invoke**. A Success message appears.

### Viewing the .CSR File

Since the .cer file is in PEM format, you can open the file using a text editor.

## Signing a CSR Using Either a Valid CA or OpenSSL

After you generate the CSR, you must sign the CSR using one of two methods. You can either:

- Sign the CSR using a well-known CA, (for example, VeriSign)

  OR

- Sign the CSR using Open SSL.

This section describes how to sign the CSR using either method.

> **Note:** If your network requires a “trusted” certificate, then follow the instructions below to sign the CSR using a valid, well-known CA.

### Using a Certification Authority to Sign the CSR

You get the signed X.509 certificate from a valid CA, such as VeriSign. The CA issues a certificate stating and guaranteeing that the key contained in the certificate belongs to the person or organization noted in the certificate. The CA verifies the identify of the applicant's so that users can trust certificates issued by that CA to belong to the people and data identified in it, and not to an imposter.

**Certificate Formats:**

The ME certificate file can be in the following formats:

- PKCS#12: Public Key Cryptography Standard #12 format from Microsoft IIS Version 5 (binary)
- PEM: Privacy-enhanced mail (PEM) encoded format from any OpenSSL-based Web server (ASCII)

### Using OpenSSL to Sign the CSR

This section provides information on how you can generate a self-signed certificate for testing TLS with the ME using OpenSSL. This is an alternative method to using a valid CA to sign the CSR.

This section describes how to do the following things:

- Create an OpenSSL Certificate Authority (CA).
- Generate a private key and CSR on the ME system and sign in with the OpenSSL CA.
- Generate a Private Key and CSR without the ME system (not supported).
- Use OpenSSL to convert an X.509 certificate and/or RSA key to a Public-Key Cryptography Standard #12 (PKCS#12) format.

> **Note:** Before using this method, download the OpenSSL program and install it on a Unix/Linux or Windows system. You also need to add the location of the OpenSSL executables to the PATH. In a Windows environment, this will need to do this manually, requiring a reboot to take effect.

#### Creating an OpenSSL Certificate Authority

To create and Open SSL Certificate Authority (CA) on a Unix/Linux system, perform all steps as "root." On a Windows system, perform all steps as "Administrator."

1. Create directories to store certificates.

   The main CA folder is the directory where the Certificate Authority files will reside. The "private" directory stores the private keys. The "certs" directory stores the certificates (or public keys). The "csrs" directory stores the Certificate Signing Requests.

   **On Unix:**

   ```
   mkdir /CA
   mkdir /CA/private
   mkdir /CA/csrs
   mkdir /CA/certs
   ```

   **Windows:**

   ```
   mkdir C:\CA
   ```

```
mkdir C:\CA\private
mkdir C:\CA\csrs
mkdir C:\CA\certs
```

2. Create files to support the generation process.

   Create the “index.txt” file with no contents. This is the database to which OpenSSL keeps track of generated certificates generated. Create the “serial” file with a number so that each generated certificate is labeled with a number for tracking purposes.

   **Unix:**

```
touch /CA/index.txt
echo 01 > /CA/serial
```

   **Windows:**

```
copy con C:\CA\index.txt
echo 01 > C:\CA\serial
```

3. Create the OpenSSL configuration file.

   **Note:** The following are example values only.

   **Unix:**

   Using a text editor, create "/CA/openssl.cnf."

```
[ ca ]
default_ca = local_ca
[ local_ca ]
dir = /CA
certificate = $dir/certs/ca.cer
database = $dir/index.txt
new_certs_dir = $dir/certs
private_key = $dir/private/ca.key
serial = $dir/serial
default_crl_days = 365
default_days = 365
default_md = md5
policy = local_ca_policy
x509_extensions = local_ca_extensions
[ local_ca_policy ]
commonName = supplied
stateOrProvinceName = optional
countryName = optional
emailAddress = optional
organizationName = optional
organizationalUnitName = optional
[ local_ca_extensions ]
basicConstraints = CA:true
nsCertType = server
[ root_ca_extensions ]
basicConstraints = CA:true
nsCertType = server
[ req ]
default_bits = 2048
default_keyfile = /CA/private/ca.key
default_md = md5
prompt = yes
distinguished_name = root_ca_distinguished_name
```

```
x509_extensions = root_ca_extensions
[ root_ca_distinguished_name ]
countryName = Country Name (2 letter code)
countryName_default = US
countryName_min = 2
countryName_max = 2
stateOrProvinceName = State or Province Name (full name)
stateOrProvinceName_default = MA
localityName = Locality Name (eg, city)
localityName_default = Maynard
0.organizationName = Organization Name (eg, company)
0.organizationName_default = Acme Packet, Inc.
organizationalUnitName = Organizational Unit Name
(eg,section)
organizationalUnitName_default = Support
commonName = Common Name (eg, YOUR name)
commonName_max = 64
emailAddress = Email Address
emailAddress_default = jgentile@acmepacket.com
emailAddress_max = 64
[ req_attributes ]
challengePassword = A challenge password
challengePassword_min = 4
challengePassword_max = 20
unstructuredName = An optional company name
```

**Windows:**

Using a text editor, create "C:\CA\openssl.cnf."

**Note:** The following are example values only.

```
[ ca ]
default_ca      = local_ca

[ local_ca ]
dir             = C:\\CA
certificate     = $dir\\certs\\ca.cer
database        = $dir\\index.txt
new_certs_dir   = $dir\\certs
private_key     = $dir\\private\ca.key
serial          = $dir\\serial

default_crl_days        = 365
default_days            = 365
default_md              = md5

policy          = local_ca_policy
x509_extensions = local_ca_extensions

[ local_ca_policy ]
commonName              = supplied
stateOrProvinceName     = optional
countryName             = optional
emailAddress            = optional
organizationName        = optional
organizationalUnitName  = optional

[ local_ca_extensions ]
basicConstraints        = CA:false
```

```
nsCertType              = server

[ root_ca_extensions ]
basicConstraints        = CA:true
nsCertType              = server

[ req ]
default_bits    = 2048
default_keyfile = C:\\CA\\private\\ca.key
default_md      = md5

prompt                  = yes
distinguished_name      = root_ca_distinguished_name
x509_extensions         = root_ca_extensions

[ root_ca_distinguished_name ]
countryName                     = Country Name (2 letter code)
countryName_default             = US
countryName_min                 = 2
countryName_max                 = 2

stateOrProvinceName             = State or Province Name (full name)
stateOrProvinceName_default     = MA

localityName                    = Locality Name (eg, city)
localityName_default            = Maynard

0.organizationName              = Organization Name (eg, company)
0.organizationName_default      = Acme Packet, Inc.

organizationalUnitName          = Organizational Unit Name (eg,
section)
organizationalUnitName_default  = Support

commonName                      = Common Name (eg, YOUR name)
commonName_max                  = 64

emailAddress                    = Email Address
emailAddress_default            = jgentile@acmepacket.com
emailAddress_max                = 64

[ req_attributes ]
challengePassword               = A challenge password
challengePassword_min           = 4
challengePassword_max           = 20

unstructuredName                = An optional company name
```

4. Generate the CA's private key and Master Certificate (public key).

   This generates two files:

   - CA/private/ca.key (C:\CA\private\ca.key on Windows) – This is the CA's private key used to sign certificates. Keep this secure. If this key is compromised, it can be used to create certificates for malicious purposes.
   - CA/certs/ca.cer (C:\CA\certs\ca.cer on Windows) – This is the CA's certificate (public key). This is the file that would be distributed to client's "Trusted Root" stores to trust any certificates signed by this CA's private key.

     **Unix:**

     ```
     openssl req -x509 -new -config /CA/openssl.cnf –days 3000 -out /CA/
     ```

```
certs/ca.cer
```

**Windows:**

```
openssl req -x509 -new -config C:\CA\openssl.cnf –days 3000 -out
C:\CA\certs\ca.cer
```

The ca.key is created automatically based on the configuration file.

Enter a strong passphrase for the CA key. Remember it, as this helps protect the security of your CA.

Fill in the following fields:

```
Country Name (2 letter code) [US]: <your country>
State or Province Name (full name) [MA]: <your state/province>
Locality Name (eg, city) [Maynard]: <your locale>
Organization Name (eg, company) [Acme Packet]: <your company>
Organizational Unit Name (eg, section) [Support]: <your department>
Common Name (eg, YOUR name) []: <Use the FQDN of the CA system
running OpenSSL>
Email Address []: <your email address>
```

**Note:** The **common-name** field on the "cert-gen" page is the most important. This is the name that is used to validate the certificate. Use the Fully Qualified Domain Name (FQDN) of the appropriate ME system, such as abc.123example.com.

5. Change permissions on the CA's key to only allow "root" access:

   **Unix:**

   ```
   chmod 700 /CA/private/ca.key
   ```

   **Windows:**

   ```
   echo y|cacls C:\CA\private\ca.key /G %COMPUTERNAME%\Administrator:F
   ```

   **Note:** You only need to complete the process for setting up the CA once, while the processes for signing Certificates must be repeated every time a certificate needs to be generated.

### Generating a Private Key and Certificate Signing Request Without the ME

Instead of generating the private key and CSR on the ME, you can generate it using OpenSSL exclusively. This is not the supported method.

1. Create a CSR and Private key for the ME system.

   **Unix:**

   ```
   openssl req -new -config /CA/openssl.cnf -out /CA/csrs/cxc_csr.pem
   -keyout /CA/certs/cxc_pk.pem
   ```

   **Windows:**

   ```
   openssl req -new -config C:\CA\openssl.cnf -out
   C:\CA\csrs\cxc_csr.pem -keyout C:\CA\certs\cxc_pk.pem
   ```

   Use the OpenSSL "req" utility to generate a Self-Signed Certificate (private key) and the Certificate Signing Request (CSR) in PEM format. In this example, the file names are cxc_pk.pem for the private key, and cxc_csr.pem for the CSR.

   Enter a pass phrase for the CA key, and complete he following fields:

   ```
   Country Name (2 letter code) [US]: <your country>
   ```

```
State or Province Name (full name) [MA]: <your state/province>
Locality Name (eg, city) [Maynard]: <your locale>
Organization Name (eg, company) [Acme Packet]: <your company>
Organizational Unit Name (eg, section) [Support]: <your department>
Common Name (eg, YOUR name) []: <Use the FQDN of the CXC>
Email Address []: <your email address>
```

**Note:** The **common-name** field is the most important entry. This is the name that will be used to validate the certificate. Use the FQDN of the appropriate ME system, such as abc.123example.com.

Currently, some phones, such as Eyebeam do not support wildcard certificates where **common-name** uses an asterisk character (*) in the domain name, such as *.example.com.

2. Sign the CSR with your OpenSSL CA.

   **Unix:**

   ```
   openssl ca -config /CA/openssl.cnf -in /CA/csrs/cxc_csr.pem -out /
   CA/certs/cxc.pem
   ```

   **Windows:**

   ```
   openssl ca -config C:\CA\openssl.cnf -in C:\CA\csrs\cxc_csr.pem
   -out C:\CA\certs\cxc.pem
   ```

   Enter the pass phrase for the CA key, then respond **y** to the questions to generate and commit.

3. Merge the Private key and signed Public key into one file.

   **Unix:**

   ```
   cat /CA/certs/cxc.pem /CA/certs/cxc_pk.pem > /CA/certs/cxc.list.pem
   ```

   **Windows:**

   ```
   copy /CA/certs/cxc.pem + /CA/certs/cxc_pk.pem /CA/certs/
   cxc.list.pem
   ```

4. Upload the newly generated cxc.list.pem file back to the ME, then configure a TLS certificate, as covered earlier in this chapter. Be sure to associate it with the SIP protocol on the appropriate network interface.

**Note:** You can use the /CA/certs/ca.cer (C:\CA\certs\ca.cer on Windows) file to import into a "Trusted Root Store." For example, you can install this in Windows (Internet Explorer) for use with Soft Phones, such as Eyebeam. If you deploy the ca.cer file to multiple systems into the "Trusted Root Store", then those systems will "trust" any certificates signed by this CA.

### Using OpenSSL to Convert X.509 and RSA Keys

This section describes how to use OpenSSL to convert an X.509 certificate and/or RSA key to a Public-Key Cryptography Standard #12 (PKCS#12) format.

#### Requirements

You must have a working installation of the OpenSSL software and be able to execute OpenSSL from the command line.

Refer to "CTX106627 - How to Install the OpenSSL Toolkit," for more information on obtaining and installing OpenSSL.

The PKCS#12 specifies a portable format for storing and transporting certificates, private keys, and miscellaneous secrets. It is the preferred format for many certificate handling operations and is supported by most browsers and recent releases of the Windows family of operating systems. It has the advantage of being able to store the certificate and corresponding key, root certificate, and any other certificates in the chain in a single file.

To convert X.509 and RSA keys:

1. Ensure that the certificate(s) and key are in PEM format.
   - To convert a certificate from DER to PEM:

     ```
     x509 –in input.crt –inform DER –out output.crt –outform PEM
     ```
   - To convert a key from DER to PEM:

     ```
     rsa –in input.key –inform DER –out output.key –outform PEM
     ```
   - To convert a key from NET to PEM:

     ```
     rsa –in input.key –inform NET –out output.key –outform PEM
     ```

   **Note:** The obsolete NET (Netscape server) format is encrypted using an unsalted RC4 symmetric cipher so a passphrase will be requested. If you do not have access to this passphrase it is unlikely you will be able to recover the key

2. Use the **openssl** command to read the PEM encoded certificate(s) and key and export to a single PKCS#12 file as follows:

   ```
   openssl pkcs12 -export -in input.crt -inkey input.key -out
   bundle.p12
   ```

   **Note:** By default, the key will be encrypted with Triple DES so you will be prompted for an export password (which may be blank).

   The PEM formatted root certificate and any other certificates in the chain can be merged into a single file such as root.crt, and included in the PKCS#12 file as follows:

   ```
   openssl pkcs12 -export -in input.crt -inkey input.key -certfile
   root.crt -out bundle.p12
   ```

## Updating the Self-Signed Certificate

The **cert-update** action allows you to load the signed certificate that you receive from the CA. Once you have received the file, perform the following steps:

1. Upload the file to the ME using the **Tools/Upload file** function to browse for CA's certificate. Specify the destination path on the ME system, such as /cxc/certs, and specify the destination name of the certificate.
2. Select the **Keys** tab and select the appropriate key from the Key Stores list to display the Manage Key Store page.
3. Click **Update** to browse for the file that you uploaded.
4. Click **Update** to load the signed certificate to the CXC.

   If you choose to update the certificate using the **cert-update** action rather than from the **Keys** tab, complete the fields as follows:

- **keyFile:** Specify the name and directory path of the key that you want to update.

  Example: /cxc/certs/<*myNetworkKey*>.p12
- **alias:** Specify the alias for the keyFile name, if previously created.
- **password:** Specify the password associated with the keyFile, as specified previously.
- **certFile:** Specify the name and directory path of the signed certificate that you received from the CA and uploaded to the ME using the ME Management System Tools/Upload File function or other file transfer mechanism.

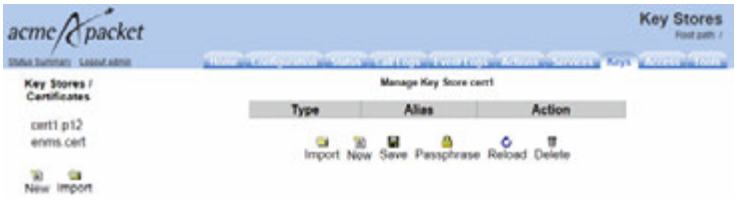
## Subject Alternative Name for HTTPS Certificates Support

The ME supports Subject Alternative Name (SAN) for use with HTTPS certificates. SAN is a X509 version 3 certificate extension that allows one to specify a list of host names protected by a single SSL certificate.

To add multiple SANs to a certificate:

1. Select the **Keys** tab and either click **New** to create a new key store or select the existing store on which you want to add a certificate.

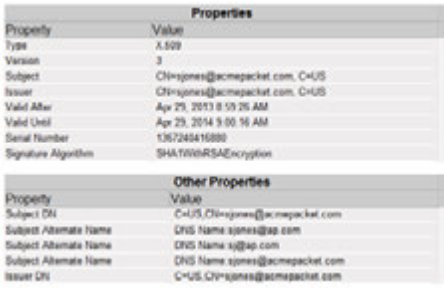

2. Enter a name and passphrase if creating a new key store and click **Create**. The key store appears.




3. Click **New**. The Generate New Self-Signed Certificate in Key Store X page appears.
4. Click **Add** beside the **Alternate name** field to add alternate host names be added to the certificate's **subjectAltName** field.
5. Click **Create**. The certificate appears in the key store.

> **Note:** When configuring the ME via the CLI, separate multiple SAN entries using the '|' character.

To view the SANs within a certificate click View next to the certificate name. The following image shows three SANs.



| Properties | |
|---|---|
| Property | Value |
| Type | X.509 |
| Version | 3 |
| Subject | CN=sjones@acmepacket.com, C=US |
| Issuer | CN=sjones@acmepacket.com, C=US |
| Valid After | Apr 29, 2013 8:59:26 AM |
| Valid Until | Apr 29, 2014 9:00:16 AM |
| Serial Number | 1367240416880 |
| Signature Algorithm | SHA1WithRSAEncryption |

| Other Properties | |
|---|---|
| Property | Value |
| Subject DN | C=US,CN=sjones@acmepacket.com |
| Subject Alternate Name | DNS Name:sjones@ap.com |
| Subject Alternate Name | DNS Name:s@ap.com |
| Subject Alternate Name | DNS Name:sjones@acmepacket.com |
| Issuer DN | C=US,CN=sjones@acmepacket.com |

# Configuring the Certificate on the Media Engine

Once you have imported the certificate to a directory on the ME system, configure the settings that control how the ME uses the certificate.

**CLI Session**

The following CLI session sets the directory and certificate destination file name path, specifies the passphrase.

```
NNOS-E>config vsp
config vsp>config tls
config tls>config certificate myNetworkCert.pfx
Creating 'certificate myNetworkCert.pfx'
config certificate myNetworkCert.pfx>set allow-null-cipher enabled
config certificate myNetworkCert.pfx>set passphrase-tag pass
```

By default, the ME only supports SSLv3 or TLSv1. If you require SSLv2 for interoperability, set this property true. Specify the passphrase-tag associated with the certificate file. Use this property if the certificate file is encrypted to have its private key information protected. This passphrase-tag must match the string with which the certificate was encrypted.

## Displaying the Certificates Installed on the Media Engine

Use the **show certificates** to command to display the list of installed certificates on the system.

# Deploying the Load Factor Application

The following sections describe the steps you must take to deploy the ME load factor application.

## About the Load Factor Application

To balance request loads between SE nodes and ME nodes, you must deploy a custom load factor application on each ME node. The load factor application reports an appropriate cluster based load factor to a SE, and SE uses that load factor to choose which ME to relay requests to. SE favors less heavily loaded instances for new requests, balancing the load among multiple ME nodes.

## About Load Factor Application Virtual Host Deployment Scenarios

There are two ways you can configure the load factor application virtual host:

- Host name virtual hosting
- IP virtual hosting

When configuring hostname virtual hosting, you assign an IP address to the load factor web service application, and you provide a Domain Name System (DNS) hostname for the virtual host.

When configuring IP virtual hosting, you assign an IP address to the load factor web service and another IP address to the virtual host.

**Note:** Hostname virtual hosting is the recommended ME configuration scheme.

## Configuring Host Name Virtual Hosting

To configure hostname virtual hosting:

1. Navigate to the ME login page:

   `https://`*`hostname`*
   The login page appears.
2. Enter your administration **Username** and **Password**, and click **Login**.

   The ME home page appears.
3. Select the **Configuration** tab.
4. Expand the **cluster** node and select the **interface** node of the **box** you want to configure.
5. In the ip row of the configuration table, click **Add ip**.
6. Enter a **name** for the Web service interface, configure the **ip-address** information as required, and click **Create**.
7. Specify the HTTP transmission **type**, HTTP or HTTPS, as well as the Web service **port** number, and click **Create**.
8. In the virtual-host row of the configuration table, click **Add virtual-host**.
9. Enter the DNS hostname you have configured for the virtual host, make sure **admin** is set to **enabled**, and click **Create**.

Continue to "Configuring the Virtual Host web-app-config Object".

## Configuring IP Name Virtual Hosting

To IP name virtual hosting:

1. Navigate to the ME login page:

   `https://`*`hostname`*
   The login page appears.
2. Enter your administration **Username** and **Password**, and click **Login**.

   The ME home page appears.
3. Select the **Configuration** tab.
4. Expand the **cluster** node and select the **interface** node of the **box** you want to configure.
5. In the ip row of the configuration table, click **Add ip**.
6. Enter a **name** for the Web service interface, configure the **ip-address** information as required, and click **Create**.

7. Select the **interface** node again.
8. In the ip row of the configuration table, click **Add ip**.
9. Enter a **name** for the Web service interface, configure the **ip-address** information as required, and click **Create**.
10. Select the **ip** object you created for the Web service, and in the web-service row of the Configuration table click **Configure**.
11. Specify the HTTP transmission **type**, HTTP or HTTPS, as well as the Web service **port** number, and click **Create**.
12. In the virtual-host row of the configuration table, click **Add virtual-host**.
13. Enter the IP address you have assigned to the virtual host, make sure **admin** is set to **enabled**, and click **Create**.

    Continue to "Configuring the Virtual Host web-app-config Object".

## Configuring the Virtual Host web-app-config Object

Once you have created and configured the ip objects, you must add a web-app-config object that points to the load factor Web Archive (WAR) file, to the virtual host.

To configure the virtual host's web-app-config object:

1. Log in to the ME console using a secure shell (SSH), and copy **loadfactor.war** from the **/cxc/ws/**samples directory to the **/cxc_common/webapps** directory.
2. Navigate to the ME login page:

   `https://`*`hostname`*
   The login page appears.
3. Enter your administration **Username** and **Password**, and click **Login**.

   The ME home page appears.
4. Select the **Configuration** tab.
5. Expand the **cluster** node and select the **interface** node of the **box** you want to configure.
6. Expand the **ip** object you created, expand **web-service**, and select the **virtual-host** object.
7. In the web-app-config row of the configuration table, click **Add web-app-config**.
8. Enter the path to the load factor application, **/cxc_common/webapps/loadfactor.war** and click **Create**.
9. In the context-parameter row of the configuration table, click **Add context-parameter**.
10. Enter **meMgmtHost** for the **name**, and the DNS name or IP address of the Web service for the **value**, and click **Create**.
11. In the context-parameter row of the configuration table, click **Add context-parameter**.
12. Enter **meMgmtUsername** for the **name**, and the hostname or IP address of the Web service for the **value**, and click **Create**.

## Configuring Media Engine Communication with Signaling Engine

To enable communication between the ME and SE in your WebRTC Session Controller installation you must add the MEs to your SE configuration and configure the ME callback which specifies the load balancer endpoint for ME nodes.

### Adding Media Engines to Signaling Engine

To add MEs to SE:

1. Start your SE servers if they are not already running. See "Starting the Signaling Engine Servers" for more information.
2. Navigate to the WebRTC Session Controller SE console and log in with your administrator username and password:

   `http://hostname:port/wsc-console`

   **Note:** The default SE console port is 7001.

3. Select the **Configuration** tab.
4. Click **Lock and Edit**.
5. In the ME pane, enter the following information:
   - **User:** Enter the ME administrative username.
   - **Password:** Enter the password for the administrative username.
6. Click the **Add** button and enter the following information:
   - **Address:** Enter the hostname or IP address of the ME Node.
   - **Port:** Enter the port of the ME Node.
7. Click **OK** to save the ME Node. Add additional nodes as required.
8. Click **Commit** to save your changes.

For more information on the ME options, see *Oracle Communications WebRTC Session Controller System Administrator's Guide*.

### Configuring the Media Engine Callback

To configure the ME callback:

1. Start your SE servers if they are not already running. See "Starting the Signaling Engine Servers" for more information.
2. Navigate to the WebLogic Server administration console and log in with your administrator user name and password:

   `http://hostname:port/console`

   **Note:** The default administration console port is 7001.

3. In the Domain Structure pane, expand **Environment**, and select **Servers**.
4. In the Summary of Servers pane, select the **Configuration** tab.
5. Select your SE server from the Servers table.

**Note:** You must repeat this procedure for each SE in a clustered environment.

6. In the Settings for the SE, select the **Protocols** tab.
7. Select **wsc-me-callback** from the Network Channels table.
8. If you need to override the default values for **Listen Address** and **Listen Port**, uncheck **Enabled**.
9. Update the following information:
   - **Listen Address** and **Listen Port**: Enter the hostname or IP address and the port of the SE which will serve as the primary endpoint for the ME HTTP callbacks. You can only update this field if **Enabled** is unchecked.
   - **External Listen Address** and **External Listen Port**: Enter the hostname or IP address and port of the load balancer or the SE which will serve as the backup endpoint for the ME HTTP callbacks if the primary endpoint cannot be reached.
10. If you have modified the default values for **Listen Address** or **Listen port**, check **Enabled**.
11. Click **Save**.
12. Log out of the administration interface.

### Configuring Media Engine Anchoring

Table 10–1 describes the media anchoring options supported by WebRTC Session Controller.

***Table 10–1 WebRTC Session Controller Routing Options***

| Scenario | Description |
|---|---|
| Web to Web conditional anchoring | Dynamic Media Anchoring (DMA) is enabled. Browsers are allowed to stream media between each other directly. If they cannot directly reach each other (for instance if they are behind a Network Address Translation (NAT) firewall and no Traversal Using Relays around NAT (TURN) service is configured), media is relayed through ME. Calls from WebRTC endpoints to and from non-WebRTC/SIP/PSTN endpoints are automatically supported. |
| Web to Web forced anchoring | DMA is enabled and all media is routed through ME. Calls from WebRTC endpoints to and from non-WebRTC/SIP/PSTN endpoints are automatically supported. |
| Web to Web bypassing ME | DMA is disabled, and nothing is passed through ME. This should only be done for diagnostic purposes. |

You can modify the anchoring behavior by modifying constants in the SE GroovyScript library.

To change the SE to ME anchoring scheme:

1. Start your SE servers if they are not already running. See "Starting the Signaling Engine Servers" for more information.
2. Navigate to the WebRTC Session Controller SE console and log in with your administrator user name and password:

```
http://hostname:port/wsc-console
```

**Note:** The default SE console port is 7001.

3. Select the **Script Library** tab.
4. Click **Lock and Edit**.
5. Modify the script library as required:
   - For Web to Web conditional anchoring, set:

     ```
     public static final DMA_ENABLED = true
     public static final ME_CONFIG_NAME_DMA = web-to-web-anchor-conditional
     ```
   - For Web to Web forced anchoring, set:

     ```
     public static final DMA_ENABLED = true
     public static final ME_CONFIG_NAME_DMA = web-to-web-anchored
     ```
   - To bypass ME instances for diagnostic purposes:

     ```
     public static final DMA_ENABLED = false
     ```

   **Note:** When DMA_ENABLED is set to false, the value of ME_CONFIG_NAME_DMA does not matter.
6. Click **Validate Library** to ensure you introduced no errors. Fix any errors that are reported.
7. Click **Commit** to save your changes.

# 11

# Installing Media Engine Clusters

This chapter provides information on how to install a Media Engine (ME) cluster, a group of ME systems that operate together to support redundancy and failover, high-availability, load balancing, and configuration.

## Media Engine Cluster Overview

A "high-availability" cluster is a group of ME systems that provides a single point of configuration management, and at the same time, expands functionality across multiple devices participating in the cluster. An ME *master* manages the configuration for the entire cluster. All members of the cluster share network resources, network load, media ports and streaming, registration, and other processes.

ME systems within a cluster may be geographically dispersed in the network. A cluster recovers from the failure of one or more cluster members through health monitoring, shared master services migration, and network redundancy using the Virtual Router Redundancy Protocol (VRRP).

A cluster can be set up to operate as a two-system primary/standby redundant configuration.

## Cluster Operations and Services

In the two-system redundant configuration, one ME system is the active master, performing signaling & media processing, and the other ME system is available as a standby system for the signaling & media processing if the master fails. Master failover allows another ME system to assume the master role in the cluster should the originally configured master become unavailable. VRRP is responsible for handling the failover from the master to the backup device.

### Master-Services

The **master-services** configuration is responsible for mirroring the state of the cluster to allow reliable failover to a standby device.

---

**Note:** You must have the **master-services > events** object enabled for clustering to work.

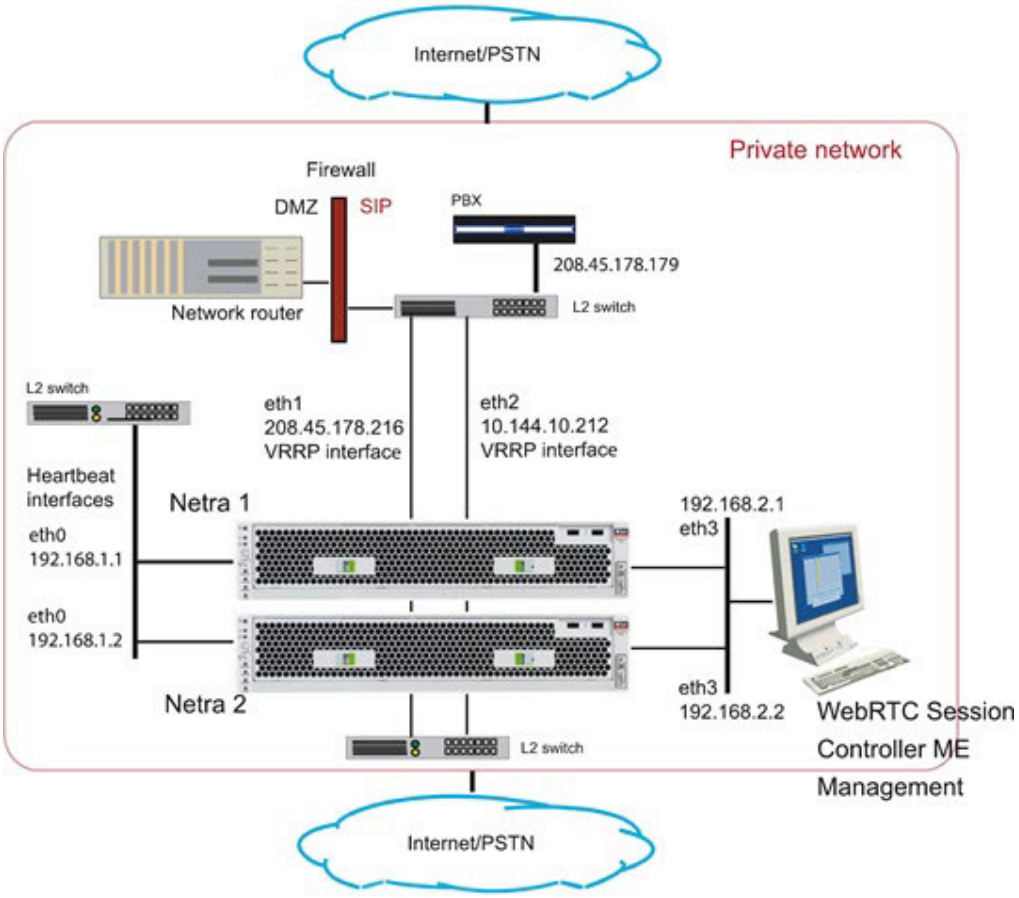
---

The following sections describe the suggested settings for the **master-services** objects:

### Cluster-Master

A **cluster-master** configuration on the ME system designated as the master is responsible for passing configuration changes to cluster members. A secondary property called **takeover-timer-value** specifies the number of milliseconds (such as 500) that the master-service stays in "awaiting takeover" mode at boot time.

Use the **show cluster-master -v** command to display the **current takeover-timer** value. When the ME boots, each hosted master-service waits for this period to determine if any existing devices in the cluster are already running that service before assuming mastership.




### Accounting

When enabled, accounting services supports RADIUS accounting, system logging (syslog), DIAMETER protocol services, the accounting database, and the accounting file-system.

### Database

The master-services **database** object allows you to configure maintenance and other settings for the ME system database. The ME database is the local repository for call accounting records and media files.

The **database** master service should be on a backup ME system, with the secondary property **preempt** set to *true*. This will help maintain the data in one location in the event of a brief service outage.

The **preempt** property specifies whether the master-service should resume the mastership if it has gone down and then returned to operation. If set to *true*, the master resumes its position. If set to *false*, the backup service retains master control.

### Server-Load

The master-services **server-load** object configures the ME to calculate server load. This object must be enabled if your dial plan arbiter rule settings use **least-load** as the routing algorithm option. (The arbiter rule property sets the criteria by which the ME selects the server to which it forwards calls.)

Configure the **server-load** master-service for outbound server load balancing or server based admission\emission control. Currently, the **server-load** master-service should be linked to the VRRP SIP signaling interfaces over a configured group.

### Call-Failover

The **call-failover** master-service configures failover for the media and signaling streams. As a master-service, the configured host ME master distributes copies of the media and kernel rules to all backup devices in a cluster. The ME uses the database on the host box, but enabling **call-failover** ensures that there is an active copy of the database on another device in the cluster in the event of a failure.

### Load-Balancing

The master-services **load-balancing** object configures ME systems to host the load-balancing master service. For detailed information, see Configuring Cluster Load Balancing.

### File-Mirror

The master-services **file-mirror** object sets all participating ME systems to share particular files (the types of files shared are preset in the ME operating system), such as media recordings, log files, etc. The file-mirror master service distributes files to all ME systems listed as hosts for the service.

Once the files are mirrored, you can play them back from any ME system that functions as a host.

### Sampling

The master-services **sampling** object opens the mechanism for setting the interval at which the ME samples operational aspects of the system for either:

- Display in the ME Management System, or
- For sending to a server

By setting sampling for a status provider, you can view data for that provider over a specified period of time. The ME supports two sampling targets: a Postgres SQL database and an IBM tivoli server. (Set the provider data sent to the target using the **status** and **provider** objects. See *Oracle Communications WebRTC Session Controller Media Engine Object Reference* for more information on configuring these objects.)

Once you have enabled **sampling**, the master service stores the samples in its local database.

## Heartbeat Interface, BOOTP, and Messaging

Use the Ethernet physical interface **eth0** as the heartbeat interface for the ME cluster. This interface is used by default for any backup ME system that you added to the

cluster. The systems will perform a BOOTP request over that interface and you will be able to add these systems by creating an entry for each to the configuration, and then booting them.

Once an ME is a member of the cluster, that system will receive a saved configuration file (*cxc.cfg)* from the master. Each time the *cxc.cfg* file is saved on the master, the latest copy of the cxc.cfg file is sent to each device in the cluster. You will need to configure a messaging interface on each cluster member so that the master knows the interface over which members of the cluster will receive the cxc.cfg file.

## Event Logging

Event logs are stored on each box individually and represent the events that occurred on that particular ME system. You configure event logging in the **services/event-logs** configuration object. The recommended event log filters on a cluster are as follows:

- Local-database all error
- File *name* system error
- File *name* krnlsys info
- File *name* system info
- File *name* db info

## Network Time Protocol (NTP)

Ensure that you have NTP configured on all ME systems, ensuring that they point to a timeserver which will keep their time synchronized.

**WARNING: DO NOT use a VRRP interface as your route to the timeserver, since one device will always have the VRRP interfaces down and will not be able to contact the NTP server.**

If you do not have access to an external NTP server, configure one of the clustered ME systems to be an NTP server for the other cluster members. It is important to run NTP, as the time on all clustered system must be kept synchronized. If the times on the ME systems drift apart, the Denial of Service (DOS) software will not function properly, as timestamps are required to make this work across the cluster.

You can configure the NTP-server on the messaging interface on one ME system, and have all other devices point to this IP address in their NTP-client configuration.

## Cluster Redundancy Operations

The ME cluster redundancy operates as follows:

- Internal messaging is exchanged so that each ME system knows the state of the other boxes, either up or down.
- If the active cluster master goes down, the box listed next in the list of cluster masters becomes the active cluster master. (Note that control does not automatically go back to the original system when it returns to service.)
- All the other master services work similarly, with an ordered list of devices that can run the service and the active service running on the next device in the list if the active master fails.

If an ME system fails, another device in the cluster will assume its network interfaces using VRRP.

### Notes on Cluster Management

The ME cluster management operates as follows:

- Within a given cluster, one box functions as the active cluster master.
- Configuration and management of all boxes within a cluster is performed through the cluster master.
- There are no limitations on how many boxes within the cluster can be configured as backup cluster masters or backups for any of the master services.
- The configuration contains a list of boxes that can be cluster masters. The ordering of this list reflects the order in which boxes attempt to become master (for example, the box listed first becomes the initial master, if that box fails then the next box in the list attempts to become the master.)
- The ME Management System connects to the cluster master and provides a single point of management for the following:
  - Configuration
  - Status reports
  - Call logs
  - Accounting data
  - Actions
- The CLI provides single point of management for configuration using the CLI on the cluster master. The CLI is still available on all the other devices in the cluster, so any CLI commands can be executed on individual boxes.
- Note that the management functionality available from a given cluster is dependent on the functionality being performed by that cluster. For example, call logs are available only on clusters where signaling is performed; media recordings are available only on clusters where media streaming is performed.

## Cluster Installation Prerequisites

Before beginning the cluster installation, ensure that any L2/L3 switch supporting the cluster has the Port Fast, Fast Link, or similar feature turned on. This allows the switch to run the Spanning Tree 802.1 protocol so that the switch ports being used by the ME go directly to the "forwarding" state. If the switch does not support Port Fast or Fast Link, disable the Spanning Tree protocol for the VLANs associated with the switch ports being used by the ME.

## Cluster Installation Procedure

There are a number of steps that you need to follow to install an ME network cluster. You will need to know certain information about all the systems in the cluster for proper operation.

Each step uses a sample CLI session of commands that best illustrate how to best configure important settings.

1. Determine the specific ME system to assume the role of cluster master. Configure **master-services** to specify the device the cluster to assume initial mastership.

```
NNOS-E> config master-services
config master-services> config cluster-master
config cluster-master> set admin enabled
config cluster-master> set host-box cluster\box 1
config cluster-master> set host-box cluster\box 2
config cluster-master> set group 1
config cluster-master> return
```

2. Note the MAC address (identifier) on each device in the cluster. The MAC address is on a sticker on the back of the system. Make a note of each MAC address.

   On each device, if there is no sticker present, attach a laptop or standard PC to the system console port and perform the following steps:

   - Power up the system
   - At the NNME prompt, execute the **show interface-details eth0** command to display the MAC address.

3. Attach a console to the cluster master and power up the ME system.

4. Configure the cluster master by configuring the Ethernet interfaces, IP addresses, and protocols. Ethernet interface eth0 is the "heartbeat" interface for the cluster. Use the eth0 interface on each ME system as the connection to the cluster.

```
NNOS-E> config cluster
config cluster> config box 1
config box 1> set identifier 00:04:23:d7:9f:34
config box 1> config interface eth0
config interface eth0> config ip heartbeat
Creating 'ip heartbeat'
config ip heartbeat> set ip-address static 192.168.1.1/24
config ip heartbeat> config telnet
config telnet> return
config ip heartbeat> config ssh
config ssh> return
config ip heartbeat> config bootp-server
config bootp-server> return
config ip heartbeat> config vrrp
config vrrp> return
```

**Note:** Optionally, you can run the config setup script to configure the IP addresses, management port, and other settings presented in the script.

By configuring messaging on the ME master, the master looks through the configurations of all other devices to find out which interface is used for messaging. (If multiple interfaces are configured, the master only communicates with the first one it finds.) The master then communicates with the identified interface to share configuration and data.

```
config ip heartbeat> config messaging
config messaging> set admin enabled
config messaging> set certificate vsp tls
certificate 208.45.178.216.pfx
config messaging> set port 5312
config messaging> set protocol tls
config messaging> return
config ip heartbeat> return
config interface eth0> return
```

```
config box 1>
```

Configure the interface and the protocols over which you will run management sessions to the ME. This is an "out-of-band" interface that allows you to separate management traffic from SIP signaling and media streams.

```
config box 1> config interface eth3
Creating 'interface eth3'
config interface eth3> config ip mgmt
Creating 'ip mgmt'
config ip mgmt> set ip address static 192.168.2.1/24
config ip mgmt> config ssh
config ssh> return
config ip mgmt> config web
config web> set protocol https 443 0
config web> return
config ip mgmt> config sip
config sip> set udp-port 5060
config sip> return
config ip mgmt> config icmp
config icmp> return
config ip mgmt> config media-ports
config media-ports> return
config ip-mgmt> return
config interface eth3> return
config box 1> config cli
config cli> set prompt nn2610-1
config cli> set banner ""
config cli> set display paged 50
config cli> return
config box 1> return
config cluster>
```

5. Configure the second ME system in the cluster. Note that you also configure eth0 as the "heartbeat" interface to the cluster.

```
config cluster> config box 2
config box 2> set hostname nn2610-2
config box 2> set name ""
config box 2> set contact ""
config box 2> set location ""
config box 2> set identifier 00:04:23:c3:22:f4
config box 2> config interface eth0
config interface eth0> config ip heartbeat
config ip heartbeat> set ip-address static 192.168.1.2/24
config ip heartbeat> config telnet
config telnet> return
config ip heartbeat> config ssh
config ssh> return
config ip heartbeat> config web
config web> set protocol https 443 0
config web> return
config ip heartbeat> config icmp
config icmp> return
config ip heartbeat> config vrrp
config vrrp> return
config ip heartbeat> config messaging
config messaging> set admin enabled
config messaging> set certificate vsp tls
certificate 208.45.178.216.pfx
config messaging> set port 5312
```

```
config messaging> set protocol tls
config messaging> return
config ip heartbeat> return
config interface eth0> return
config box 2>
```

Configure the interface and the protocols over which you will run management sessions. This is an "out-of-band" interface that allows you to separate management traffic from SIP signaling and media streams.

```
config box 2> config interface eth3
Creating 'interface eth3'
config interface eth3> config ip mgmt
Creating 'ip mgmt'
config ip mgmt> set ip address static 192.168.2.2/24
config ip mgmt> config ssh
config ssh> return
config ip mgmt> config web
config web> set protocol https 443 0
config web> return
config ip mgmt> config sip
config sip> set udp-port 5060
config sip> set nat-translation enabled
config sip> return
config ip mgmt> config icmp
config icmp> return
config ip mgmt> config media-ports
config media-ports> return
config ip-mgmt> return
config interface eth3> return
config box 1> config cli
config cli> set prompt NNOS-E-2
config cli> set banner ""
config cli> set display paged 50
config cli> return
config box 1> return
config cluster> set share media-ports true
config cluster> set share signaling-entries true
config cluster> set mirror-media-streams true
```

6. Configure VRRP on the ME interfaces to handle the public and private sides of the network. Note that the first VRRP interface connects the public side; the second VRRP interface connects the private side.

   A VRRP configuration for IP interfaces includes a list of box/interface pairings. The first pair in this list is the *primary interface*. The second pair in the list is the *backup interface* and will take over if the primary goes down. You can configure additional levels of redundancy by specifying more box/interface pairs of lower priority. Priority is based on the positioning of the **set host-interface** command.

```
config cluster> config vrrp
config vrrp> config vinterface vx0
config vinterface vx0> set group 1
...vinterface vx0> set host-interface cluster box 1 interface eth1
...vinterface vx0> set host-interface cluster box 2 interface eth1
config vinterface vx0> config ip public
Creating 'ip public'
config ip public> set ip-address static 208.45.178.216/28
config ip public> config ssh
config ssh> return
config ip public> config web
```

```
config web> set protocol https 443 0
config web> return
config ip public> config sip
config sip> set nat-translation enabled
config sip> set udp-port 5060
config sip> set tcp-port 5060
config sip> set tls-port 5061
config sip> set certificate vsp\tls\certificate 208.45.178.216.pfx
config sip> return
config ip public> config icmp
config icmp> return
config ip public> config media-ports
config media-ports> return
config ip public> config routing
config routing> config route default
Creating 'route default'
config route default> set gateway 208.45.178.209
config route default> return
config routing> return
config ip public> return
config vinterface vx0> return
config vrrp>

config cluster> config vrrp
config vrrp> config vinterface vx1
config vinterface vx1> set group 1
...vinterface vx1> set host-interface cluster box 1 interface eth2
...vinterface vx1> set host-interface cluster box 2 interface eth2
config vinterface vx1> config ip private
Creating 'ip private'
config ip private> set ip-address static 208.45.178.216/28
config ip private> config ssh
config ssh> return
config ip public> config web
config web> set protocol https 443 0
config web> return
config ip private> config sip
config sip> set nat-translation enabled
config sip> set udp-port 5060
config sip> set tcp-port 5060
config sip> set tls-port 5061
config sip> set certificate vsp\tls\certificate 208.45.178.216.pfx
config sip> return
config ip private> config icmp
config icmp> return
config ip private> config media-ports
config media-ports> return
config ip private> config routing
config routing> config route static-to-asx
Creating 'route static-to-asx'
config route static-to-asx> set destination network 208.45.178.0/24
config route static-to-asx> set gateway 10.144.10.254
config route static-to-asx> return
config routing> return
config ip private> return
config vinterface vx1> return
config vrrp> return
config cluster> return
```

7. Configure the master-services that you want to run on the cluster.

```
config> config master-services
config master-services> config accounting
config accounting> set host-box cluster\box 1
config accounting> set host-box cluster\box 2
config accounting> set group 1
config accounting> return
config master-services> config database
config database> set host-box cluster\box 1
config database> set host-box cluster\box 2
config database> set group 1
config database> set media enabled
config database> return
config master-services> return
config>
```

8. For TLS, you will need to upload the TLS certificate file on each ME system in the cluster. Copy the certificate that you receive from the CA to the ME using a compatible file transfer mechanism, such as PuTTY Secure Copy (PSCP). If you have the file on a local network PC, use PSCP to move the file to a directory path on the ME.

   The following example PSCP command copies the certificate file named **208.45.178.216.pfx** from the PC root directory to the ME system at IP address **208.178.216.pfx** in the directory **/cxc/certs/208.45.178.216.pfx**.

   ```
   C:\ pscp -l root -pw sips -P 2200 208.45.178.216.pfx
   208.45.178.216:/cxc/certs/208.45.178.216.pfx
   ```

   The following CLI session sets the directory and certificate file name path, specifies the passphrase, and whether to allow SSL Version 2 operability.

   ```
   NNOS-E> config vsp
   config vsp> config tls
   config tls> config certificate 208.45.178.216.pfx
   config certificate 208.45.178.216.pfx> set allow-sslv2 true
   config certificate 208.45.178.216.pfx> set certificate-file
   /cxc/certs/208.45.178.216.pfx.pfx
   config certificate 208.45.178.216.pfx> set passphrase-tag pass
   ```

   By default, the ME only supports SSLv3 or TLSv1. If you require SSLv2 for interoperability, set this property **true**. Specify the passphrase-tag associated with the certificate file. Use this property if the certificate file is encrypted to have its private key information protected. This passphrase tag must match the string with which the certificate was encrypted.

9. Power up the other ME systems in the cluster and connect them to the network. This initiates a configuration download from the cluster master so the systems acquire their initial configuration (IP addresses, etc.).

10. Use the CLI or ME Management System at the cluster master to configure any additional features. These features include the objects and settings under the VSP object, including:

    - default-session-config
    - registration-plan
    - dial-plan
    - enterprise servers, carriers, and gateways

# Configuring External Messaging

Messaging is the mechanism the ME uses to communicate among boxes in a cluster. Messaging sets up a listening socket on an interface, enabling the interface to receive messaging traffic and participate in clustering and media partnering.

In a cluster, the master looks through the configurations of all ME systems to find out which interface is used for messaging. (If multiple interfaces are configured, the master only communicates with the first one it finds.) The master then communicates with the identified interface to share configuration and data.

In media partnering, you configure a specific IP address (on a different box) as a partner. On the box that owns that IP address, you need to configure and enable messaging for media partnering to operate.

## CLI Session

Example 11–1 configures messaging on box 1, interface eth0.

***Example 11–1 Configuring Messaging On a Box and Interface***

```
NNOS-E> config cluster
config cluster> config box 1
config box 1> config interface eth0
config interface eth0> config ip boston1
config ip boston1> config messaging
config messaging> set admin enabled
config messaging> set certificate vsp tls certificate
name
config messaging> set port 13002
config messaging> set protocol tls
```

# Configuring Cluster Load Balancing

Load balancing of SIP processing across cluster interfaces requires both headend and backing interfaces. The *headend* interface is the central distribution point. It does not do any SIP processing, it only forwards the calls to its configured backing interfaces. When you configure a SIP phone, you would configure it to point to the headend interface.

To configure an IP interface as a headend interface, you simply configure the **sip** object with backing interfaces. Their presence contained within the IP configuration results in the interface being treated by the ME as a headend interface.

The *backing-interfaces* are identified as such within this **sip** object. In the **backing-interface** property, you reference previously configured IP interfaces. The backing interface is the location at which the ME terminates TCP and TLS connections (and where UDP transport messages arrive) and handles SIP processing. The ME uses round-robin load-balancing to distribute message across the configured backing interfaces.

To correctly configure load-balancing for SIP processing, you must do the following:

1. Configure the IP interfaces that will be used for both the headend and backing interfaces.
2. The SIP properties of the backing interfaces must match those of the head interface. For example, they must all use the same port assignments, and if you are using TLS, they must all use the same certificate.

3. You must enable the **master-services registration** object so that the interfaces can share the registration database.

To verify your configuration, first ensure that all SIP properties match. From the CLI at the headend, execute the **show load-balance** command. This lists all associated backing interfaces (and statistics). From each box hosting a backing interface, execute **show backing-interface** to display configuration and statistics information.

Example 11–2, "Configuring Load Balancing SIP Traffic" assumes that you have configured a three-box cluster, with box 1 containing the headend interface, with boxes 2 and 3 containing the backing interfaces over which traffic is load balanced. This session sets the backing interfaces for load balancing SIP traffic that is distributed from the headend interface at IP address 215.2.3.0/24.

## CLI Session

***Example 11–2 Configuring Load Balancing SIP Traffic***

```
config> config cluster
config cluster> config box 1
config box 1> config interface eth1
config interface eth1> config ip public
Creating 'ip public'
config ip public? set ip-address static 215.2.3.0/24
config ip public> config sip
config sip> config load-balancing
config load-balancing> set backing-interface cluster box 2 interface eth1 ip
public
Creating 'cluster\box 2\interface eth1\ip public'
config load-balancing> set backing-interface cluster box 3 interface eth1 ip
public

config sip> show
cluster
 box 1
  interface eth1
   ip public
    sip
     admin enabled
backing-interface cluster\box 2\interface eth1\ip public2
     backing-interface cluster\box 3\interface eth1\ip public3


NNOS-E> show load-balance

Head-end IP 215.2.3.0:  undersubscribed:
-------------------------------------------------------------------------------
Backing IP      State       Added   Removed   Maximum   Current  Percent
-------------------------------------------------------------------------
215.6.7.0       Down            0         0         0         0     0.0%
215.8.9.0       Down            0         0         0         0     0.0%
-------------------------------------------------------------------------------
Totals:                         0         0         0         0   100.0%

NNOS-E>
```

## Restarting a Media Engine Cluster

You can perform a simultaneous warm restart of all systems in a cluster by using the **restart cluster** command. A warm restart simply restarts the ME applications on each system without rebooting the operating system.

If you warm restart an individual device in the cluster, the ME automatically rejoins the cluster when it comes back up. If that box is hosting a master service or a VRRP interface, the service or interface may fail over to a different ME system.

If you need to shut a system down by turning the power off, use the **restart halt** command before pressing the power button or disconnecting the power source. A **restart halt** will properly prepare a system for a shutdown. The ME system will rejoin the cluster when it comes back up.

# 12

# Configuring Secure Media (SRTP) Sessions

This chapter provides information on configuring inbound and outbound encryption on Session Initiation Protocol (SIP) media sessions anchored by the Media Engine (ME).

## Anchoring Media Sessions

Media anchoring forces the SIP media session to traverse the ME system. The **auto** setting enables conditional anchoring where the ME uses its auto-anchoring algorithms to determine anchoring necessity based on a variety of criteria, including whether you have configured smart anchoring via the **autonomous-ip** object and whether the calling devices are behind a firewall.

## Configuring Inbound and Outbound Encryption

For secure inbound and outbound media sessions, you need to configure ME **in-encryption** and **out-encryption** settings. Inbound encryption handles the portion of the call from the initiator to the ME using a specified encryption method. Similarly, outbound encryption handles the portion of the call from the ME to the call recipient using a specified encryption method.

### Inbound Encryption Mode and Type

Set the inbound encryption mode to one of the following settings:

- **none**: The ME disables the encryption put forth by the incoming endpoint. (That is, it responds "no" to the encryption portion of the authentication handshake.) If the outbound endpoint requires encryption, then the call is dropped.
- **allow**: The ME passes the call through, leaving the encryption setting unchanged.
- **require**: The call must come in with encryption specified or the ME drops it.

Set the inbound encryption type to one of the following settings:

- **RFC-3711**: Use encryption as defined in RFC 3711, The Secure Real-time Transport Protocol (SRTP). This is the same encryption as used in the ME setting.

### Outbound Encryption Mode, Type, and Require-TLS Setting

Set the out-encryption mode to one of the following settings:

- **none**: The ME disables the encryption put forth by the outbound endpoint. (That is, it responds "no" to the encryption portion of the authentication handshake.) If the inbound endpoint requires encryption, then the call is dropped.

- **offer**: The ME changes or establishes the encryption type to the value specified in the **type** property, below.
- **follow**: If the inbound endpoint offered encryption, the ME offers that type to the outbound endpoint.
- **require**: The call must come in with encryption specified or the ME drops it.

Set the out-encryption type to one of the following settings:

- **RFC-3711**: Use encryption as defined in RFC 3711, The Secure Real-time Transport Protocol (SRTP). This is the same encryption as used in the ME setting.

> **Note:** Because the ME does not always know on the outbound leg the encryption method expected by the recipient (because that recipient isn't in the registry), you must manually set the type of encryption to offer.

### Require TLS

The **require-tls** property specifies the requirements of the signaling protocol for a call's outbound leg. That is, it defines whether the ME offers SRTP over a non-secure (TCP or UDP) signaling connection. The action of this property depends on the setting of the mode property. When this property is set to:

- **true**: The ME only offers encryption when talking to a TLS client. If TLS and SRTP are required (**mode** is set to **require**), the ME fails calls going to TCP/UDP clients. If the mode property is set to **offer** or **follow**, the ME forwards the call without SRTP.
- **false**: The ME offers SDP messages according to the mode setting without regard for the signaling transport. This allows keys to be exchanged in an insecure message.

Most phones follow *RFC 4568, SDP Security Descriptions for Media Streams*, https://tools.ietf.org/html/rfc4568, and thus require that this property be set to *true.*

# 13

# Creating and Commissioning USB Sticks

This chapter provides information on creating and commissioning Media Engine (ME) USB software installation sticks for commissioning third-party servers.

As part of each download, and depending on your actual requirements, Oracle can provide the following:

- USB Boot Media Creator (BMC) with the ME software
- Documentation on how to create a USB stick and commission the ME software on your selected hardware
- Standard set of Oracle ME technical publications

## Supported USB Sticks

You must provide a USB stick with at least 1GB storage, and up to 4 GB storage, to handle ME software downloads. Oracle has tested a variety of USB sticks available from current suppliers and manufacturers. Most 1GB USB sticks manufactured today will work.

## USB Stick Restrictions

Files that are larger than 2 MB will not be backed up to the USB stick and restored during the upgrade process.

All *.cfg and *.xml files in the current working directory (/cxc) that are less than 2 MB in size are backed up to the stick and restored during the upgrade.

---

**Note:** Oracle recommends manually backing up the config file locally prior to the upgrade process.

---

## Important Note About the New USB Stick

The ME USB stick provides three important functions:

- **Commissioning**: Boots and licenses a new third-party server using the ME software.
- **Rescue utilities**: After you have successfully commissioned the system (booted and licensed), the system automatically rewrites the USB stick so that you can use it to run system utilities in the event of a catastrophic failure. You will not be able to use this USB stick again to license another system. The USB stick can only be used at the specific system from which it was originally written.

Additionally, licensing information is rewritten to the USB stick, directly associating the license with the system. Use the **show system-info** action to display the box identifier (box-id) to which this USB stick is associated. The USB stick also contains log and debug files that you can use to help diagnose problems associated with the USB licensing process.

- **Rescue stick creation**: With the original USB commissioning stick, use the **restore-stick-create full-backup** action to capture the current software, certificates, and operating system image to the USB stick. Oracle recommends that you use **restore-stick-create full-backup** to preserve the image prior to performing a system software upgrade, or whenever you have made significant and reliable changes to the system configuration.

## Creating a New USB Rescue Stick

The **restore-stick-create full-backup** action allows captures the current ME software and the operating system image and creates a new USB rescue stick. Oracle recommends that you use **restore-stick-create full-backup** to preserve the image prior to performing a system software upgrade, or whenever you have made significant and reliable changes to the system configuration.

Perform the following steps:

1. At the NNOS-E prompt, type **umount usb**.

   Ignore the warning about the USB stick not being mounted.
2. Remove the USB stick and wait at least five seconds before reinserting the stick.
3. Invoke the **restore-stick-create full-backup** action. The resulting USB is a boot device from which you can restart and restore the ME.

   **Note:** The log event indicating that the operation has completed successfully appears while data is being written to the stick. DO NOT immediately remove the USB stick when you see this log event. Instead, issue the umount usb command again, and wait for it to complete.
4. Remove the restore USB when the **restore-stick-create** action has completed.

   **Note:** PostgreSQL database, media recordings, system tar files(.gz) are not written to the restore stick with the restore-stick-create action.

   **Note:** Use the restore-stick-create config-backup action to create a restore stick containing the current configuration file only.

## Using the Rescue Utility USB

Perform the following steps:

1. Insert the rescue utility USB into one of the USB slots.
2. At the NNOS-E prompt, if available, enter **restart cold** to do a full restart.

3. The USB and disk lamps will blink during the boot-up process followed by a series of system messages. Press any key to continue, or perform the appropriate action below. On the
   - **Sun Netra X5-2:** Press **F8**.
   - **Sun Netra X3-2:** Press **F8**.
   - **HPDL360 G7:** Press **F11**.
   - **HPDL585 G7:** Press **F11**.
   - **HPDL320 G8:** Press **F11**.
   - **HPDL360 G8:** Press **F11**.
   - **HPDL160 G9:** Press **F11**.
   - **CXC-350**: Press the **ESC** key.
   - **Sun x4150**: Press **F8**.
   - **IBM X3650, X3550, X-3350**: Press **F12**.
   - **Dell 1950 and 2950**: Press **F11**.
   - **Dell R220**: Press **F11**.
   - **IBM HS20 and HS21**: Press **F12**.
   - **Fugitsu Siemens RX200 S5:** Press **F12**.
   - **Fugitsu Siemens RX300 S4:** Press **F12**.
   - **ATCA MCLB0040**: Press **F8**.
4. The system enters utility mode and the following menu appears:


You can perform one of the following:

- Rescue an Oracle Linux system

> **Note:** This boots the system into rescue mode and attempts to mount the ME drive under /mnt/sysimage.).

- Install Oracle Linux 7.2.
- Test this media and install Oracle Linux 7.2.
- Install Oracle Linux 7.2 in basic graphics mode.

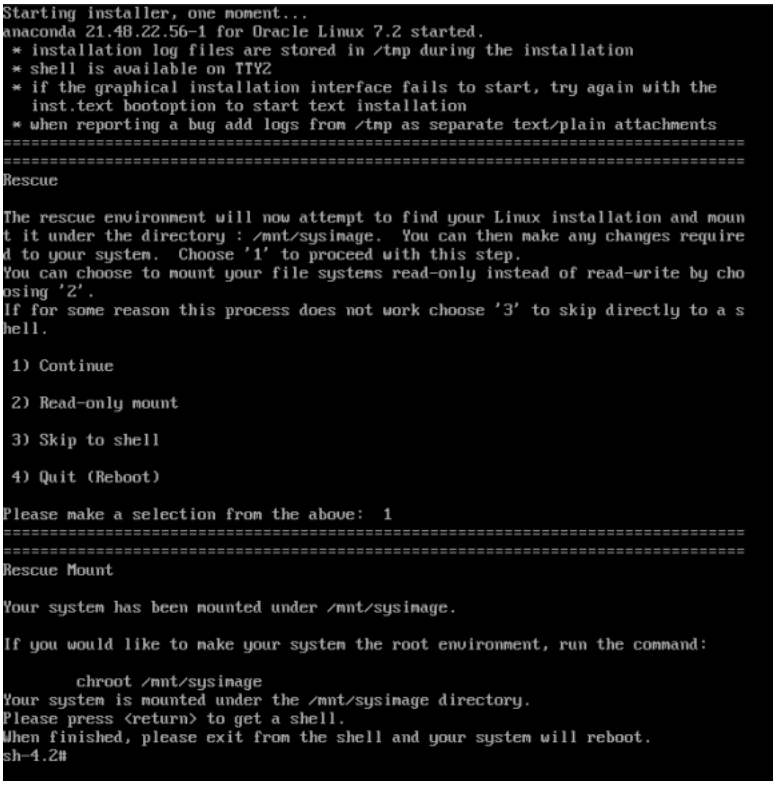
**Note:** The above three options run the Oracle Linux 7.2 installer. These actions wipe out any existing configuration and data. Use either the **restore-stick-create** action or the rescue mode, ensure your configuration and other data files have been backed up. Once Oracle Linux is reinstalled, the ME packages must be reinstalled. For more information on installing the ME, see <blue>Installing the Media Engine.

- Run a memory text.
- Boot from the local drive. (Boots from the local hard disk and defaults to the first disk. To change the disk, use the **<Tab>** key.

## Using the Expert Mode

The Expert mode provides utilities that allow you to recover from system failures where there is apparent damage to the software and configuration file, and where recovery is necessary to return the ME to normal operation.

If you enter rescue mode, the system boots and attempts to mount the ME hard drive at /mnt/sysimage. If there are multiple Oracle Linux installations, there is a prompt to select which installation.




If the system mounts the drive successfully, you can attempt to repair it or perform a backup.

### Backing Up the Configuration

Execute the save_box_config.sh script to copy the configuration.

> **Note:** The USB drive must be mounted prior to executing this script.

```
sh-4.2# mkdir /mnt/usb
sh-4.2# mount /dev/disk/by-label/ASC_RESCUE /mnt/usb
sh-4.2# save_box_config.sh ?mnt/sysimage /mnt/usb
sh-4.2# umount /mnt/usb
```

After a fresh Oracle Linux and ME installation, booting with this USB stick inserted automatically restores this configuration.

# 14

# Installing and Running the ME Virtual Machine

This chapter provides information on downloading, installing, and running the ME Virtual Machine (ME VM) software in virtual OS environments. This software is the same software as used for non-virtual OS but has been packaged specifically as ME VM for use in virtual OS environments.

> **Note:** If you are installing a patch set for the Media Engine, refer to the *Release Notes* and the *ReadMe* files that accompany the patch set for information on installing it.

The ME virtual machine is designed to be used as an evaluation platform so that potential customers can test the ME software in an environment that does not require them to install the software on a dedicated piece of hardware. In some cases, the virtual machine can also be used in production environments provided that the customer understands the limitations associated with using the ME VM software in a virtual OS environment.

## Server-Based Requirements

Before downloading the virtual machine to an x86-based server, ensure that the VM host meets the following hardware and software requirements:

- x86-based Windows or Linux server with Intel 32- or 64-bit dual-core processors
- 2GB minimum (4 GB recommended) physical memory for each VM instance
- Minimum of 40GB hard disk space per VM instance
- One or two Ethernet interfaces
- OVM 3.3.1, VMware ESXi 5.5, and Xen 3.4.3

## Linux Installations

If you are installing the ME Virtual Machine on a Linux workstation running VMware, Oracle recommends the following technical resources:

- For Server 1.0: http://www.vmware.com/support/pubs/server_pubs.html
- For Player 1.0 and 2.0: http://www.vmware.com/support/pubs/player_pubs.html

# Installing the VM

This section describes installing WebRTC Session Controller in a virtual machine environment.

## Installing the Media Engine on an Oracle Virtual Machine

The ME is certified to run on the Oracle Virtual Machine (OVM) 3.3.1.

### Prerequisites

You must meet the following prerequisites before installing the ME on an OVM.

- A Network File System (NFS) has been mounted for VM storage with an additional storage file server for repository
- A server pool has been created
- Server (s) have been discovered and added to this pool
- The ISO file has been imported
- Networks and Virtual MAC range have been created
- VM Console access (VNC) has been made available

You create the ME VM via the OVM Manager GUI. The OVM Manager binds to the weblogic server on the Oracle Linux host's 7002 SSL port.

Access the OVM Manager using the following link:

`https://x.x.x.x:7002/ovm/console`

Where *x.x.x.x* is the OVM Manager's IP address.

1. Log into the OVM Manager using the user name and password configured when you set up the OVM.
2. Create external routable interfaces by selecting the **Networking** tab, selecting the **Networks** button, and clicking the plus (+) icon.
3. Create a new bridge **bonds/ports only** and select **Virtual Machine** in the **Network Uses** field.
4. Bind the new bridge to a free port on the VM host.
5. *Optional*. Create a heartbeat interface (if you choose to configure clustered VMs) by selecting the **Networking** tab, selecting the **Networks** button, and clicking the plus (+) icon.
6. Create a new bridge local network only and select **Virtual Machine** in the **Network Uses** field.
7. Create MAC addresses for each Virtual NIC by selecting the **Networking** tab, selecting the **Virtual NICs** button, and creating a **Dynamic MAC Address Range**.

   **Note:** You must create a unique MAC address for each Virtual NIC.

   You must mount an NFS to host the ME.

   When creating a VM, your storage repository contains an ISO file and the new VM immediately boots from the Virtual DVD.

To boot the VM from the Virtual DVD:

1. Select the **Repositories** tab and select the repository you created from the **File Server**.
2. Select **ISOs**.
3. Via HTTP, import the ME's .iso code.

   You are now ready to create a VM.

To create a VM:

1. Select the Servers and VMs tab and choose the server on which you are hosting the VM.
2. Select **Create Virtual Machine** and click **Next**.
3. Specify a **Name** and set the **Memory** and **Processors** for this VM and click **Next**.

   **Note:** The default **Memory** is **1024** and the default number of **Processors** is **1**.

4. Select your networks and click **Next**.

   **Note:** The order you select the networks affects how the ME Ethernets align.

   These MAC addresses (whether assigned dynamically or statically) now appear as assigned MAC addresses under the Virtual NICs tab in OVM Manager.

To create the VM virtual disk:

1. Select the Virtual Disk's **Disk Type**, select the **Create a Virtual Disk** icon, and click **Next**.
2. Select the previously-created **Repository** and enter a **Virtual Disk Name** and a **Size** (Oracle recommends 40 GB) and click **OK**.

To point the ME ISO code to commission the VM:

1. Select **CD/DVD** from the Slot 2 **Disk Type** drop-down menu, select **Select an ISO**, and click **Next**.
2. Select the previously-imported ISO and click **OK**.

Once the ME ISO code is pointed to commission the VM, set the Boot Options. The first time you boot, you utilize the CDRom as nothing resides on the Disk yet. All subsequent boots utilize the Disk and ignore the CDRom.

**Note:** If you choose CDRom as the first boot option, the initial boot, as well as all subsequent boots, continue to utilize the CDRom.

To set the Boot Options:

1. Select **Disk**.
2. Select **CDRom** and click **Finish**.

   To see the newly-created VM, select the **Servers and VMs** tab and click **Virtual Machines** from the **Perspective** drop-down menu. At this point in the installation process, the **Status** of the VM is **Stopped**.

To start the VM:

1. Select **Start** to start the VM.
2. Select Launch Console.

   The OVM Console now displays the installation process.
3. Type **y** and press <*Enter*> when prompted to complete the installation process.

   The VM reboots and once the installation is complete you see the ME login prompt. The ME is now ready to be set up and configured.

## Configuring OVM Passthrough

On the OVM, there are two ways to directly connect a VM to a physical port: Single Root I/O Virtualization (SR-IOV) and Peripheral Component Interconnect (PCI) Passthrough. You configure hardware passthrough at the OVM Server's CLI.

> **Note:** Prior to configuring hardware passthrough, you must have a fully built VM, however, any NICs designated for hardware passthrough may not have an associated Network.

SR-IOV is a specification that treats a single physical device as multiple separate Virtual Functions (VF)s.

> **Note:** In development, SR-IOV was found to be available on 10GB ixgbe devices only.

To configure SR-IOV:

1. Access and log into the OVM Server's CLI.
2. Install the necessary packages on the OVM server. Example 14–1 shows an example installation.

***Example 14–1 Installing Packages On OVM***

```
libibumad-1.3.8-2.mlnx1.5.5r2.el5.x86_64.rpm
libibmad-1.3.9-7.mlnx1.5.5r2.el5.x86_64.rpm
opensm-libs-3.3.15-6.mlnx1.5.5r2.el5.x86_64.rpm
kernel-ib-1.5.5.092-2.6.39_300.29.1.el5uek.x86_64.rpm
infiniband-diags-1.5.13.MLNX_20120708-4.mlnx1.5.5r2.el5.x86_64.rpm
ovsvf-config-1.0-6.noarch.rpm
```

3. Create a python script called **vnfs.py** to view and marry PCI addresses to interfaces. Example 14–2 shows an example python script.

***Example 14–2 Python Script***

```
#!/usr/bin/python
# Copyright (C) 2012 Steve Jordahl
#
# This program is free software; you can redistribute it and/or modify
# it under the terms of the GNU Lesser General Public License as published
# by the Free Software Foundation; version 2.1 only.
#
# This program is distributed in the hope that it will be useful,
# but WITHOUT ANY WARRANTY; without even the implied warranty of
```

```
# MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
# GNU Lesser General Public License for more details.
#
# vfns: list SR-IOV virtual functions

import os

info = {}

def catFile(filename):
        readfile = open(filename)
        return readfile.read().strip()
for dev in os.listdir('/sys/class/net'):
        if dev.startswith('eth'):
                info[dev] = {}
                info[dev]['address'] = catFile('/sys/class/net/' + dev +
'/address')

for dev in info.keys():
        devLink = os.readlink('/sys/class/net/' + dev + '/device')
        info[dev]['pci address'] = devLink[-7:]
        os.chdir('/sys/class/net/' + dev)
        for devInfo in os.listdir(devLink):
                if devInfo.startswith('virtfn'):
                        info[dev][devInfo] = os.readlink(os.path.join(devLink,
devInfo))[-7:]

for dev in sorted(info.keys()):
        print dev
        for detail in sorted(info[dev].keys()):
                print "     " + detail + ":  " + info[dev][detail]
```

4. Create **/etc/pciback/pciback.sh**. Example 14–3 shows an example file.

***Example 14–3 Example File***

```
#!/bin/sh
if [ $# -eq 0 ] ; then
echo "Require a PCI device as parameter"
exit 1
fi
for pcidev in $@ ; do
if [ -h /sys/bus/pci/devices/"$pcidev"/driver ] ; then
echo "Unbinding $pcidev from" $(basename $(readlink
/sys/bus/pci/devices/"$pcidev"/driver))
echo -n "$pcidev" > /sys/bus/pci/devices/"$pcidev"/driver/unbind
fi
echo "Binding $pcidev to pciback"
echo -n "$pcidev" > /sys/bus/pci/drivers/pciback/new_slot
echo -n "$pcidev" > /sys/bus/pci/drivers/pciback/bind
done
```

5. Use an Input/Output Memory Management Unity (IOMMU) to allow both the VM and physical devices access to memory. The IOMMU allows the OVM to limit what memory a device is allowed and gives the device the same virtualized memory layout that the guest sees. Example 14–4 shows an example IOMMU.

***Example 14–4 IOMMU***

```
Edit /boot/grub/grub.conf to enable iommu and comment out the existing kernel
```

```
entry ( see example )

# grub.conf generated by anaconda
#
# Note that you do not have to rerun grub after making changes to this file
# NOTICE:  You have a /boot partition.  This means that
#          all kernel and initrd paths are relative to /boot/, eg.
#          root (hd0,0)
#          kernel /vmlinuz-version ro root=/dev/sdb2
#          initrd /initrd-[generic-]version.img
#boot=/dev/sdb
default=0
timeout=5
splashimage=(hd0,0)/grub/splash.xpm.gz
hiddenmenu
title Oracle VM Server-ovs (xen-4.3.0 3.8.13-26.4.2.el6uek.x86_64)
root (hd0,0)
        #kernel /xen.gz console=com1,vga com1=57600,8n1 dom0_mem=max:1776M
allowsuperpage dom0_vcpus_pin dom0_max_vcpus=20
        kernel /xen.gz console=com1,vga com1=57600,8n1 dom0_mem=max:1776M
allowsuperpage iommu=passthrough,no-qinval,no-intremap
        module /vmlinuz-3.8.13-26.4.2.el6uek.x86_64 ro
root=UUID=e2b44279-55a5-48b9-b910-82446b7b8c65 rd_NO_LUKS rd_NO_LVM LANG=en_
US.UTF-8 rd_NO_MD SYSFONT=latarcyrheb-sun16  KEYBOARDTYPE=pc KEYTABLE=us rd_NO_DM
rhgb quiet
        module /initramfs-3.8.13-26.4.2.el6uek.x86_64.img
```

6. Add SR-IOV support to ovs.conf. Example 14–5 shows SR-IOV ovs.conf support.

> **Note:** The following example configures support for 10 VFs on the server's 4 ixgbe interfaces (matching up to ethernets 9-12).

***Example 14–5 SR-IOV Ovs.conf Support***

```
[root@meads ~]# vi /etc/modprobe.d/ovs.conf
options bnx2x disable_tpa=1
options ipv6 disable=1
# SRIOV support
options ixgbe max_vfs="10,10,10,10,0,0,0,0,0,0,0,0,0,0"
install ixgbe /sbin/modprobe pciback ; /sbin/modprobe --first-time
--ignore-install ixgbe
```

7. Blacklist the Intel VF driver (ixgbevf) in dom0 so that the dom0 kernel does not try to use the VFs. Example 14–6 shows Intel VF driver (ixgbevf) blacklisted.

***Example 14–6 Intel VF Driver Blacklisted***

```
[root@Meads ~]# vi /etc/modprobe.d/blacklist.conf
#
# Listing a module here prevents the hotplug scripts from loading it.
# Usually that'd be so that some other driver will bind it instead,
# no matter which driver happens to get probed first.  Sometimes user
# mode tools can also control driver binding.
#
# Syntax:  driver name alone (without any spaces) on a line. Other
# lines are ignored.
#

# watchdog drivers
```

```
blacklist i8xx_tco

# framebuffer drivers
blacklist aty128fb
blacklist atyfb
blacklist radeonfb
blacklist i810fb
blacklist cirrusfb
blacklist intelfb
blacklist kyrofb
blacklist i2c-matroxfb
blacklist hgafb
blacklist nvidiafb
blacklist rivafb
blacklist savagefb
blacklist sstfb
blacklist neofb
blacklist tridentfb
blacklist tdfxfb
blacklist virgefb
blacklist vga16fb
# ISDN - see bugs 154799, 159068
blacklist hisax
blacklist hisax_fcpcipnp

# intel ixgbe sr-iov vf (virtual function) driver
blacklist ixgbevf
```

8. Reboot the OVM server.
9. Run the vnfs script to view addresses and VFs statistics. Example 14–7 shows the vnfs script.

***Example 14–7 Vnfs Script***

```
[root@Meads ~]# ./vnfs.py
eth0
     address:  a0:36:9f:2c:39:74
     pci address:  30:00.0
eth1
     address:  a0:36:9f:2c:39:75
     pci address:  30:00.1
eth10
     address:  00:21:28:a1:e2:41
     pci address:  88:00.1
     virtfn0:  88:10.1
     virtfn1:  88:10.3
     virtfn2:  88:10.5
     virtfn3:  88:10.7
     virtfn4:  88:11.1
     virtfn5:  88:11.3
     virtfn6:  88:11.5
     virtfn7:  88:11.7
     virtfn8:  88:12.1
     virtfn9:  88:12.3
eth11
    address:  00:21:28:a1:e2:42
     pci address:  98:00.0
     virtfn0:  98:10.0
     virtfn1:  98:10.2
     virtfn2:  98:10.4
```

```
     virtfn3:  98:10.6
     virtfn4:  98:11.0
     virtfn5:  98:11.2
     virtfn6:  98:11.4
     virtfn7:  98:11.6
     virtfn8:  98:12.0
     virtfn9:  98:12.2
eth12
address:  00:21:28:a1:e2:43
     pci address:  98:00.1
     virtfn0:  98:10.1
     virtfn1:  98:10.3
     virtfn2:  98:10.5
     virtfn3:  98:10.7
     virtfn4:  98:11.1
     virtfn5:  98:11.3
     virtfn6:  98:11.5
     virtfn7:  98:11.7
     virtfn8:  98:12.1
     virtfn9:  98:12.3
eth2
     address:  a0:36:9f:2c:39:76
     pci address:  30:00.2
eth3
     address:  a0:36:9f:2c:39:77
     pci address:  30:00.3
eth4
     address:  a0:36:9f:2d:0b:a8
     pci address:  a0:00.0
eth5
     address:  a0:36:9f:2d:0b:a9
     pci address:  a0:00.1
eth6
     address:  a0:36:9f:2d:0b:aa
     pci address:  a0:00.2
eth7
     address:  a0:36:9f:2d:0b:ab
     pci address:  a0:00.3
eth8
     address:  00:21:28:a1:e2:46
     pci address:  5f:00.0
eth9
     address:  00:21:28:a1:e2:40
     pci address:  88:00.0
     virtfn0:  88:10.0
     virtfn1:  88:10.2
 virtfn2:  88:10.4
     virtfn3:  88:10.6
     virtfn4:  88:11.0
     virtfn5:  88:11.2
     virtfn6:  88:11.4
     virtfn7:  88:11.6
     virtfn8:  88:12.0
     virtfn9:  88:12.2
```

**10.** Run the module. Example 14–8 shows running the module.

***Example 14–8 Running the Module***

```
[root@meads ~]# modprobe xen-pciback
```

**11.** Assign devices to pciback in the format:

```
Domain 0:Bus:#:Device#:Function #).
```

Example 14–9 shows an example of adding devices to pciback.

> **Note:** In the following example, the 4 interfaces are VFs on ethernets 9-12.

***Example 14–9 Adding Devices to Pciback***

```
[root@meads ~]# /etc/pciback/pciback.sh 0000:88:10.0
Unbinding 0000:88:00.0 from ixgbe
Binding 0000:88:00.0 to pciback

[root@meads ~]# /etc/pciback/pciback.sh 0000:88:10.1
Unbinding 0000:88:00.1 from ixgbe
Binding 0000:88:00.1 to pciback

[root@meads ~]# /etc/pciback/pciback.sh 0000:98:10.0
Unbinding 0000:98:00.0 from ixgbe
Binding 0000:98:00.0 to pciback

[root@meads ~]# /etc/pciback/pciback.sh 0000:98:10.1
Unbinding 0000:98:00.1 from ixgbe
Binding 0000:98:00.1 to pciback
```

**12.** View the list of VMs. Example 14–10 shows a list of configured VMs.

***Example 14–10 Configured VMs***

```
[root@meads ~]# xm list
Name                                        ID   Mem VCPUs      State   Time(s)
0004fb0000060000f9b493a2c24f9549             7  8067    16     -b----  80716.4
Domain-0                                     0  1775    20     r-----  30379.2
```

**13.** View the list of assignable devices. Example 14–11 shows a list of assignable devices.

***Example 14–11 Assignable Devices***

```
[root@meads ~]# xm pci-list-assignable-devices
0000:88:10.0
0000:88:10.1
0000:98:10.0
0000:98:10.1
```

**14.** Assign these devices to the VM. Example 14–12 shows assigning devices to the VM.

> **Note:** In the following example the VM ID is 7.

***Example 14–12 Assigning Devices to the VM***

```
[root@meads ~]# xm pci-attach 7 0000:88:10.0
[root@meads ~]# xm pci-attach 7 0000:88:10.1
[root@meads ~]# xm pci-attach 7 0000:98:10.0
```

```
[root@meads ~]# xm pci-attach 7 0000:98:10.1
```

15. View the list of devices for this VM. Example 14–13 shows devices for this VM.

***Example 14–13 Devices For This VM***

```
[root@Meads ~]# xm pci-list 7
Vdev Device
04.0 0000:88:10.0
05.0 0000:88:10.1
06.0 0000:98:10.0
07.0 0000:98:10.1
```

Now VFs on ethernets 9-12 are assigned to VM ID 7, but there are still VFs available to the host. These interfaces appear when you run the **ifconfig** command.

16. Access and log into the ME CLI and execute the **echo**, **run**, and **restart warm** commands. Example 14–14 shows the **echo**, **run**, and **restart warm** commands.

***Example 14–14 Echo, Run, and Restart Warm Commands***

```
NNOS-E>echo "1" > /sys/bus//pci/rescan
NNOS-E>run ./install_build_mactab.sh
NNOS-E>restart warm
```

After the restart has completed, these interfaces are available on the ME.

PCI Passthrough is a specification that allows you to directly connect one VM to one physical device, making the device unavailable to other VMs.

To configure PCI Passthrough:

1. Access and log into the OVM Server's CLI.
2. Install the necessary packages on the OVM Server. Example 14–15 shows installing packages on the OVM.

***Example 14–15 Installing Packages On OVM***

```
libibumad-1.3.8-2.mlnx1.5.5r2.el5.x86_64.rpm
libibmad-1.3.9-7.mlnx1.5.5r2.el5.x86_64.rpm
opensm-libs-3.3.15-6.mlnx1.5.5r2.el5.x86_64.rpm
kernel-ib-1.5.5.092-2.6.39_300.29.1.el5uek.x86_64.rpm
infiniband-diags-1.5.13.MLNX_20120708-4.mlnx1.5.5r2.el5.x86_64.rpm
ovsvf-config-1.0-6.noarch.rpm
```

3. Create a python script called **vnfs.py** to view and marry PCI addresses to interfaces. Example 14–16 shows the python script.

***Example 14–16 Python Script***

```
#!/usr/bin/python
# Copyright (C) 2012 Steve Jordahl
#
# This program is free software; you can redistribute it and/or modify
# it under the terms of the GNU Lesser General Public License as published
# by the Free Software Foundation; version 2.1 only.
#
# This program is distributed in the hope that it will be useful,
# but WITHOUT ANY WARRANTY; without even the implied warranty of
# MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
```

```
# GNU Lesser General Public License for more details.
#
# vfns: list SR-IOV virtual functions

import os

info = {}

def catFile(filename):
        readfile = open(filename)
        return readfile.read().strip()

for dev in os.listdir('/sys/class/net'):
        if dev.startswith('eth'):
                info[dev] = {}
                info[dev]['address'] = catFile('/sys/class/net/' + dev +
'/address')

for dev in info.keys():
        devLink = os.readlink('/sys/class/net/' + dev + '/device')
        info[dev]['pci address'] = devLink[-7:]
        os.chdir('/sys/class/net/' + dev)
        for devInfo in os.listdir(devLink):
if devInfo.startswith('virtfn'):
                        info[dev][devInfo] = os.readlink(os.path.join(devLink,
devInfo))[-7:]

for dev in sorted(info.keys()):
        print dev
        for detail in sorted(info[dev].keys()):
                print "     " + detail + ":  " + info[dev][detail]
```

4. Create **/etc/pciback/pciback.sh**. Example 14–17 shows an example file.

***Example 14–17 Example File***

```
#!/bin/sh
if [ $# -eq 0 ] ; then
echo "Require a PCI device as parameter"
exit 1
fi
for pcidev in $@ ; do
if [ -h /sys/bus/pci/devices/"$pcidev"/driver ] ; then
echo "Unbinding $pcidev from" $(basename $(readlink
/sys/bus/pci/devices/"$pcidev"/driver))
echo -n "$pcidev" > /sys/bus/pci/devices/"$pcidev"/driver/unbind
fi
echo "Binding $pcidev to pciback"
echo -n "$pcidev" > /sys/bus/pci/drivers/pciback/new_slot
echo -n "$pcidev" > /sys/bus/pci/drivers/pciback/bind
done
```

5. Use an IOMMU to allow both the VM and physical devices access to memory. The IOMMU allows the OVM to limit what memory a device is allowed and gives the device the same virtualized memory layout that the guest sees. Example 14–18 shows an example IOMMU.

***Example 14–18 IOMMU***

```
Edit /boot/grub/grub.conf to enable iommu and comment out the existing kernel
```

```
entry ( see example )

# grub.conf generated by anaconda
#
# Note that you do not have to rerun grub after making changes to this file
# NOTICE:  You have a /boot partition.  This means that
#          all kernel and initrd paths are relative to /boot/, eg.
#          root (hd0,0)
#          kernel /vmlinuz-version ro root=/dev/sdb2
#          initrd /initrd-[generic-]version.img
#boot=/dev/sdb
default=0
timeout=5
splashimage=(hd0,0)/grub/splash.xpm.gz
hiddenmenu
title Oracle VM Server-ovs (xen-4.3.0 3.8.13-26.4.2.el6uek.x86_64)
root (hd0,0)
        #kernel /xen.gz console=com1,vga com1=57600,8n1 dom0_mem=max:1776M
allowsuperpage dom0_vcpus_pin dom0_max_vcpus=20
        kernel /xen.gz console=com1,vga com1=57600,8n1 dom0_mem=max:1776M
allowsuperpage iommu=passthrough,no-qinval,no-intremap
        module /vmlinuz-3.8.13-26.4.2.el6uek.x86_64 ro
root=UUID=e2b44279-55a5-48b9-b910-82446b7b8c65 rd_NO_LUKS rd_NO_LVM LANG=en_
US.UTF-8 rd_NO_MD SYSFONT=latarcyrheb-sun16  KEYBOARDTYPE=pc KEYTABLE=us rd_NO_DM
rhgb quiet
        module /initramfs-3.8.13-26.4.2.el6uek.x86_64.img
```

6. Reboot the OVM Server.
7. Run the vnfs script to view addresses and VFs statistics. Example 14–19 shows the vnfs script.

***Example 14–19 Vnfs Script***

```
[root@meads ~]# ./vnfs.py
eth0
     address:  a0:36:9f:2c:39:74
     pci address:  30:00.0
eth1
     address:  a0:36:9f:2c:39:75
     pci address:  30:00.1
eth10
     address:  00:21:28:a1:e2:41
    pci address:  88:00.1
eth11
     address:  00:21:28:a1:e2:42
     pci address:  98:00.0
eth12
     address:  00:21:28:a1:e2:43
     pci address:  98:00.1
eth2
     address:  a0:36:9f:2c:39:76
     pci address:  30:00.2
eth3
     address:  a0:36:9f:2c:39:77
     pci address:  30:00.3
eth4
     address:  a0:36:9f:2d:0b:a8
     pci address:  a0:00.0
eth5
     address:  a0:36:9f:2d:0b:a9
```

```
     pci address:  a0:00.1
eth6
     address:  a0:36:9f:2d:0b:aa
pci address:  a0:00.2
eth7
     address:  a0:36:9f:2d:0b:ab
     pci address:  a0:00.3
eth8
     address:  00:21:28:a1:e2:46
     pci address:  5f:00.0
eth9
     address:  00:21:28:a1:e2:40
     pci address:  88:00.0
```

**8.** Run the module. Example 14–20 shows running the module.

***Example 14–20 Running the Module***

```
{root@meads ~}# modprobe xen-pciback
```

**9.** Assign devices to pciback in the format:

```
Domain ):Bus:#:Device#:Function #).
```

Example 14–21 shows assigning devices to pciback.

**Note:** In the following example, the 4 interfaces are VFs on ethernets 9-12.

***Example 14–21 Assigning Devices to Pciback***

```
[root@meads ~]# /etc/pciback/pciback.sh 0000:88:00.0
Unbinding 0000:88:00.0 from ixgbe
Binding 0000:88:00.0 to pciback

[root@meads ~]# /etc/pciback/pciback.sh 0000:88:00.1
Unbinding 0000:88:00.1 from ixgbe
Binding 0000:88:00.1 to pciback

[root@meads ~]# /etc/pciback/pciback.sh 0000:98:00.0
Unbinding 0000:98:00.0 from ixgbe
Binding 0000:98:00.0 to pciback

[root@meads ~]# /etc/pciback/pciback.sh 0000:98:00.1
Unbinding 0000:98:00.1 from ixgbe
Binding 0000:98:00.1 to pciback
```

**10.** View the list of VMs. Example 14–22 shows a list of VMs.

***Example 14–22 List of VMs***

```
[root@meads ~]# xm list
Name                                        ID   Mem VCPUs      State   Time(s)
0004fb0000060000f9b493a2c24f9549             7  8067    16     -b----  80716.4
Domain-0                                     0  1775    20     r-----  30379.2
```

**11.** View the list of assignable devices. Example 14–23 shows the list of assignable devices.

***Example 14–23 Assignable Devices***

```
[root@meads ~]# xm pci-list-assignable-devices
0000:88:00.0
0000:88:00.1
0000:98:00.0
0000:98:00.1
```

**12.** Assign these devices to the VM. Example 14–24 shows assigning devices to the VM.

> **Note:** In the following example the VM ID is 7.

***Example 14–24 Assigning Devices To The VM***

```
[root@meads ~]# xm pci-attach 7 0000:88:00.0
[root@meads ~]# xm pci-attach 7 0000:88:00.1
[root@meads ~]# xm pci-attach 7 0000:98:00.0
[root@meads ~]# xm pci-attach 7 0000:98:00.1
```

**13.** View the list of devices for this VM. Example 14–25 shows devices for this item.

***Example 14–25 Devices For This VM***

```
[root@Meads ~]# xm pci-list 7
Vdev Device
04.0 0000:88:00.0
05.0 0000:88:00.1
06.0 0000:98:00.0
07.0 0000:98:00.1
```

Now ethernets 9-12 are assigned to VM ID 7 and are not available to host any other VMs. They also do not show up when you run the **ifconfig** command.

**14.** Access and log into the ME CLI and execute the **echo**, **run**, and **restart warm** commands. Example 14–26 shows the **echo**, **run**, and **restart warm** commands.

***Example 14–26 Echo, Run, and Restart Warm Commands***

```
NNOS-E>echo "1" > /sys/bus//pci/rescan
NNOS-E>run ./install_build_mactab.sh
NNOS-E>restart warm
```

After the restart has completed, these interfaces are available on the ME.

## Installing the Media Engine on a VMware ESXi

The ME is certified to run on the VMware ESXi 5.5.

Oracle recommends the following configuration:

- vCPUs: 16 (16 sockets, 1 core per socket)
- RAM: 8GB
- Disk: 50G

To install the ME on a VMware ESXi:

**1.** Copy the ME's ISO file to the Datastore.

**2.** Click **Inventory**.

**3.** Create a new VM by clicking the ESXi server on the left.

4. Select **File > New > Virtual Machine** from the menu.
   - **Configuration**: Select **typical** to accept the default number of CPUs and amount of memory (1 CPU and 1GB). Select **custom** to change the default values. Click **Next**.
   - **Name and Location**: Enter a name for the VM. Click **Next**.
   - **Storage**: Select the Datastore. Click **Next**.
   - **Virtual Machine Version**: *For custom configuration only*. Select **Virtual Machine Version: 8**. Click **Next**.
   - **Guest Operating System**: Select **Linux** for **OS** and **Linux Oracle Linux 4/5/6 (64-bit)** for **Version**. Click **Next**.
   - **CPUs**: *For custom configuration only*. Select the number of sockets and number of cores/sockets. Click **Next**.
   - **Memory**: *For custom configuration only*. Select the memory size. Note the minimum, maximum, and recommended sizes for the guest OS you are using. Click **Next**.
   - **Network**: Select **3. Data Network**. Click **Next**.
   - **SCSI Controller**: *For custom configuration only*. Select **LSI Logic Parallel** (default). Click **Next**.
   - **Select a Disk**: *For custom configuration only*. Select **Create a new virtual disk**. Click **Next**.
   - **Create a Disk**: Specify the GB for disk capacity and choose **Thick Provision Lazy Zeroed** and **Store with the virtual machine**. Click **Next**.
   - **Advanced Options**: *For custom configuration only*. Check the checkbox for **SCSI (0:0)**. Ensure the **Independent** checkbox remains unchecked. Click **Next**.
   - **Ready to Complete**: Click **Finish**.
5. Right-click on the VM and select **Edit Settings...**.
   - Select **CD/DVD Drive 1**.
   - **Device Status**: Select **Connect at power on**.
   - **Device Type**: Select **Datastore ISO File** and choose **<install_release_version>_<build_number>.iso on Datastore1**.
   - Click **OK**.
6. Power on the VM by clicking the green play button.
7. Right-click the VM and select **Open Console**.

The ME is now ready to be set up and configured.

### Configuring ESXi Passthrough

On the ESXi, you can directly connect a VM to a physical port via the SR-IOV specification. SR-IOV treats a single physical device as multiple separate Virtual Functions (VF)s. To deploy SR-IOV, you must enable VFs at the host level.

T configure SR-IOV on the ESXi, you must have a NIC with an intel 82599 chipset or newer and a BIOS, both supporting SR-IOV.

The configuration for SR-IOV on the ESXi consists of two parts: first you must configure the ME's VM server, then you must assign individual VFs to specific VMs.

To configure the ME's VM server for SR-IOV:

1. Enable SR-IOV in the BIOS.
2. Ensure you have the latest drivers for your intel NIC (ixgbe) and ESXi version. See https://my.vmware.com/web/vmware/info/slug/datacenter_cloud_infrastructure/vmware_vsphere_with_operations_management/5_5#drivers_tools for more information on ESXi 5.5 drivers.
3. Install the appropriate drivers and reboot the host.
4. Log into the ESXi CLI shell and enter the following command to view a list of all NICs on the server and identify which NICs to use for SR-IOV.

   ```
   # lspci | grep -i 'ethernet\|network'
   ```

5. Specify the number of VFs you are assigning to each port by executing the following command:

   ```
   # esxcfg-module ixgbe -s max_vfs=<P1=n><P2=n><P3=n><P4=n>
   ```

where <*Px=n*> stands for the configured ports and their assigned VF values, less than or equal to 63. Assigning a value of 0 makes that port unavailable for SR-IOV.

> **Note:** The SR-IOV specification allows for you to partition the Physical Function (PF) into a particular number of VFs you can then attach to VMs. The maximum number of VFs you can create on a PF depends on the hardware you are using. Typically, for 10GbE chipsets equal to or newer than 825999, that number is 63.

6. Verify that you entered the correct values by entering the following command:

   ```
   # esxcfg-module ixgbe -g isgbe
   ```

7. Reboot the server.
8. View the list of configured VFs by either reentering the following command:

   ```
   #lspci | grep -i 'ethernet\|network'
   ```

   or accessing, via the vSphere GUI, **Host > Configuration > Advanced Settings**.

To configure a specific ME VM for SR-IOV:

> **Note:** To attach a VF to a VM, the VM version must be greater than or equal to 10.

1. Power off the VM.
2. Select **Settings > Hardware > Add**.
3. Select **PCI device** and select the VF you are adding to the VM.
4. Repeat this procedure for each VF you are adding to the VM.

> **Note:** If you are prompted to "reserve" resources, you may have to click that button for the VM to power on.

Once a VF is attached to a particular VM, you cannot attach it to any other VM.

## Installing the Media Engine as a XEN Virtual Machine

The ME is certified to run on XEN 3.4.3.

Oracle recommends the following configuration:

- vCPUs: 16 (16 sockets, 1 core per socket)
- RAM: 8 GB
- Disk: 50G

> **Note:** Oracle recommends using LVM partitions as disks.

1. Create a partition and download the XEN image from buildview into that partition. Example 14–27 creates a 50G partition.

***Example 14–27 Creating a Partition***

```
# lvcreate --size=50G --name=asc ol
```

Logical volume "asc" is created.

2. Download OL7.2 ISO image from the Oracle Software Delivery Cloud and copy the file to the /tmp directory on the server.
3. Create a config file for the VM at /etc/xen/asc/cfg. Example 14–28 shows an example config file.

> **Note:** The following is an example. Ensure you customize your config file, including changing the MAC addresses, to fit your environment.

***Example 14–28 Sample Config File***

```
#  -*- mode: python; -*-
#============================================================================
# Python configuration setup for 'xm create'.
This script sets the parameters used when a domain is created using xm create. Use
a separate script for each domain you create or set the parameters for the domain
on the XM command line.
# you can set the parameters for the domain on the xm command line.
#============================================================================

#----------------------------------------------------------------------------
# PV GRUB image file.
kernel = "/usr/lib/xen/boot/hvmloader"
builder = 'hvm'
device_model = '/usr/lib64/xen/bin/qemu-dm'

# Sets path to menu.lst
extra = "(hd0,1)/grub/menu.lst"
# can be a TFTP-served path (DHCP will automatically be run)
# extra = "(nd)/netboot/menu.lst"
# can be configured automatically by GRUB's DHCP option 150 (see grub manual)
# extra = ""

# Initial memory allocation (in megabytes) for the new domain.
#
# WARNING: Creating a domain with insufficient memory may cause out of
```

```
#          memory errors. The domain needs enough memory to boot kernel
#          and modules. Allocating less than 32MBs is not recommended.
memory = 8192

# A name for your domain. All domains must have different names.
name = "asc"

# 128-bit UUID for the domain.  The default behavior is to generate a new UUID
# on each call to 'xm create'.
#uuid = "06ed00fe-1162-4fc4-b5d8-11993ee4a8b9"

# List of which CPUS this domain is allowed to use, default Xen picks
#cpus = ""         # leave to Xen to pick
#cpus = "0"        # all vcpus run on CPU0
#cpus = "0-3,5,^1" # all vcpus run on cpus 0,2,3,5
#cpus = ["2", "3"] # VCPU0 runs on CPU2, VCPU1 runs on CPU3

# Number of Virtual CPUS to use, default is 1
vcpus = 4
cpus = "4-31" # all vcpus run on cpus >3

#---------------------------------------------------------------------
# Define network interfaces.

# By default, no network interfaces are configured.  You may have one created
# with sensible defaults using an empty vif clause:
#
# vif = [ '' ]
#
# or optionally override backend, bridge, ip, mac, script, type, or vifname:
#
# vif = [ 'mac=00:16:3e:00:00:11, bridge=xenbr0' ]
#
# or more than one interface may be configured:
#
# vif = [ '', 'bridge=xenbr1' ]

vif = [ 'mac=00:16:3E:62:F7:05, bridge=virbr0', 'mac=00:16:3E:72:C9:95,
bridge=messaging', 'mac=00:16:3E:06:57:B6, bridge=data' ]

#---------------------------------------------------------------------
# Define the disk devices you want the domain to have access to, and
# what you want them accessible as.
# Each disk entry is of the form phy:UNAME,DEV,MODE
# where UNAME is the device, DEV is the device name the domain will see,
# and MODE is r for read-only, w for read-write.

disk = [ 'phy:/dev/mapper/ol-asc,hda,w' ]

#---------------------------------------------------------------------
# Define frame buffer device.
#
# By default, no frame buffer device is configured.
#
# To create one using the SDL backend and sensible defaults:
#
# vfb = [ 'sdl=1' ]
#
# This uses environment variables XAUTHORITY and DISPLAY.  You
# can override that:
```

```
#
# vfb = [ 'sdl=1,xauthority=/home/bozo/.Xauthority,display=:1' ]
#
# To create one using the VNC backend and sensible defaults:
#
# vfb = [ 'vnc=1' ]
#
# The backend listens on 127.0.0.1 port 5900+N by default, where N is
# the domain ID.  You can override both address and N:
#
# vfb = [ 'vnc=1,vnclisten=127.0.0.1,vncdisplay=1' ]
#
# Or you can bind the first unused port above 5900:
#
# vfb = [ 'vnc=1,vnclisten=0.0.0.0,vncunused=1' ]
#
# You can override the password:
#
# vfb = [ 'vnc=1,vncpasswd=MYPASSWD' ]
#
# Empty password disables authentication.  Defaults to the vncpasswd
# configured in xend-config.sxp.

#---------------------------------------------------------------------
# Define to which TPM instance the user domain should communicate.
# The vtpm entry is of the form 'instance=INSTANCE,backend=DOM'
# where INSTANCE indicates the instance number of the TPM the VM
# should be talking to and DOM provides the domain where the backend
# is located.
# Note that no two virtual machines should try to connect to the same
# TPM instance. The handling of all TPM instances does require
# some management effort in so far that VM configration files (and thus
# a VM) should be associated with a TPM instance throughout the lifetime
# of the VM / VM configuration file. The instance number must be
# greater or equal to 1.
#vtpm = [ 'instance=1,backend=0' ]

#---------------------------------------------------------------------
# Configure the behaviour when a domain exits.  There are three 'reasons'
# for a domain to stop: poweroff, reboot, and crash.  For each of these you
# may specify:
#
#   "destroy",        meaning that the domain is cleaned up as normal;
#   "restart",        meaning that a new domain is started in place of the old
#                     one;
#   "preserve",       meaning that no clean-up is done until the domain is
#                     manually destroyed (using xm destroy, for example); or
#   "rename-restart", meaning that the old domain is not cleaned up, but is
#                     renamed and a new domain started in its place.
#
# In the event a domain stops due to a crash, you have the additional options:
#
#   "coredump-destroy", meaning dump the crashed domain's core and then destroy;
#   "coredump-restart', meaning dump the crashed domain's core and the restart.
#
# The default is
#
#   on_poweroff = 'destroy'
#   on_reboot   = 'restart'
#   on_crash    = 'restart'
```

```
#
# For backwards compatibility we also support the deprecated option restart
#
# restart = 'onreboot' means on_poweroff = 'destroy'
#                           on_reboot   = 'restart'
#                           on_crash    = 'destroy'
#
# restart = 'always'   means on_poweroff = 'restart'
#                           on_reboot   = 'restart'
#                           on_crash    = 'restart'
#
# restart = 'never'    means on_poweroff = 'destroy'
#                           on_reboot   = 'destroy'
#                           on_crash    = 'destroy'

#on_poweroff = 'destroy'
#on_reboot   = 'restart'
#on_crash    = 'restart'

#---------------------------------------------------------------------
#   Configure PVSCSI devices:
#
#vscsi=[ 'PDEV, VDEV' ]
#
#   PDEV   gives physical SCSI device to be attached to specified guest
#          domain by one of the following identifier format.
#          - XX:XX:XX:XX (4-tuples with decimal notation which shows
#                         "host:channel:target:lun")
#          - /dev/sdxx or sdx
#          - /dev/stxx or stx
#          - /dev/sgxx or sgx
#          - result of 'scsi_id -gu -s'.
#            ex. # scsi_id -gu -s /block/sdb
#                  36000b5d0006a0000006a0257004c0000
#
#   VDEV   gives virtual SCSI device by 4-tuples (XX:XX:XX:XX) as
#          which the specified guest domain recognize.
#

#vscsi = [ '/dev/sdx, 0:0:0:0' ]

#=====================================================================

# Guest VGA console configuration, either SDL or VNC
#sdl = 1
vnc = 1
vncpasswd=""
vncdisplay=10
vnclisten="0.0.0.0"
```

4. Start the VM.

```
# xl start /etc/xen/asc.cfg
```

5. Vnc to host:10 to start the OL7 installation process.

Once OL7 is installed, you can begin the ME installation. See "Installing the Media Engine" for more information on installing ME software.

## Installing the Media Engine on KVM

The ME is certified to run on KVM 1.5.3 on OL7.

Oracle recommends using the following configuration:

- vCPUs: 8
- RAM: 8GB
- Disk: 50G

> **Note:** Oracle recommends using LVM partitions as disks.

1. Install the KVM packages.

```
# yum install kvm libyirt
# yum install python-virtinst virt-top virt-manager virt-v2v virt-viewer
```

2. Use the virt-manager command to create your networks.
3. Install the ME guest by either using the following command in the CLI or via the virt-manager (right-click localhost (QEMU) and click New).

```
virt-install -n asc -r 8192 --os-type=linux --disk
/dev/mapper/ol-asc,device=disk,bus=virtio,size=50,sparse=false,format=raw -w
network=management,model=virtio -w network=messaging,model=virtio -w
network=data,model=virtio -c /mnt/install/<build_version>.iso --vcpus=8
```

# Configuring the VM

Once the VM is installed and running, you now must configure it to match the SIP application you are supporting. Since the VM does not have a pre-installed base configuration, Oracle provides the **config** configuration setup script that you can use to create a base configuration.

## Using Config Setup

For Oracle users who are familiar with ME, the *config setup* script enables the configuration on the VM to make it reachable via ICMP (ping), SSH, and HTTPS for further configuration. The script presents a set of questions to help you with the initial system configuration. The information in the script includes the following:

- Local hostname
- IP interface names and addresses
- SSH and Web access
- Default route and any additional static routes per interface for remote management
- User-defined ME

Every Oracle ME system has a minimum of two Ethernet interfaces. Any Ethernet interface on the system can be used for management traffic, however, Oracle recommends the use of eth1, as eth0 is reserved for fault-tolerant clustering with other ME systems. Management traffic is also supported on any interface that is carrying private or public network traffic. This means that it would be possible to use eth1 to carry SIP traffic and management traffic.

### CLI Session

```
NNOS-E-VM> config setup
set box\hostname: <name>
config box\interface: eth1
set box\interface eth1\ip a\ip-address: <ipAddress/mask>
config box\interface eth1\ip a\ssh (y or n)? n
config box\interface eth1\ip a\web (y or n)? y
config box\interface eth1\ip a\routing\route: <routeName>
set box\interface eth1\ip a\routing\route localGateway\gateway:
<ipAddress>
set box\cli\prompt: <newPrompt>
Do you want to commit this setup script (y or n) y
Do you want to update the startup configuration (y or n)? y
```

## Sample VM Configuration

This section describes a base configuration designed to support a standard SBC application where the VM functions with SIP endpoints and a PBX or feature server. The high-level details of this configuration are provided below and additional details are embedded in the configuration file itself at the end of this section.

- Two interfaces: one "outside" and one "inside."
- Management ports for ICMP, SSH, and HTTPS open on both interfaces.
- The IP address associated with a DNS resolver.
- SIP UDP, TCP, and TLS ports open on both interfaces.
- NAT traversal & media anchoring enabled.
- A sample gateway configuration for an attached PBX or feature server.
- A sample registration- and dial-plan for delegation of SIP traffic to the attached PBX or feature server.
- A local registration plan to support registrations and calls locally through the VM (for cases where there is no attached PBX or feature server).

> **Note:** Oracle recognizes that the items in the base configuration will not be 100% applicable to all ME VM deployments. However, by including these items in this sample configuration, new VM users can observe the configuration structure and hierarchy. Any necessary changes to this base configuration can be made using the procedures described in the Oracle manual set. See, “Using Oracle Documentation,” for more information.

Below is a copy of the base configuration. Note that any changes to the configuration should be made using the ME Management System (see, “Enabling the ME Management System”).

> **Note:** Oracle does not recommend editing the configuration file below directly, and then importing it into the VM. While the VM does support this function, it is possible to introduce syntax errors into the configuration file using this method. Modifying the configuration with the CLI or Management System prevents this possibility.

This section is unique to every VM; you do not need to edit this.

```
config cluster
 set name acmepacket-nnos-e-vm-demo
 config box 1
  set hostname acmepacket-nnos-e-vm-demo
  set name acmepacket-nnos-e-vm-demo
  set identifier 00:0c:29:c9:7a:e2
```

The IP address is configured as part of the configuration script execution.

```
config interface eth0
   config ip outside
    set ip-address static 172.30.3.128/22
config ssh
    return
    config web
    return
    config sip
     set nat-translation enabled
     set udp-port 5060 "" "" any 0
     set tcp-port 5060 "" "" any 0
     set tls-port 5061 "" "" any 0
     set certificate vsp\tls\certificate sample
    return
    config icmp
    return
    config media-ports
    return
    config routing
     config route default
      set gateway 172.30.0.1
return
    return
   return
  return
```

The following section of the configuration provides a DNS resolver entry and is configured as part of the configuration script execution. This is not required for operation but can be helpful if you want to use Fully Qualified Domain Names in the config instead of IPs.

```
config dns
  config resolver
   set server 192.168.1.3 UDP 53 100 ALL
  return
 return
return
```

The following IP is disabled; you can enable it once you change the IP to match your local network conditions.

```
config interface eth1
   config ip inside
    set admin disabled
    set ip-address static 192.168.1.2/24
config ssh
    return
    config web
    return
    config sip
     set udp-port 5060 "" "" any 0
     set tcp-port 5060 "" "" any 0
```

```
     set tls-port 5061 "" "" any 0
     set certificate vsp\tls\certificate sample
    return
    config icmp
    return
    config media-ports
    return
```

This routing config is provided as an example; edit it as needed. Change to match your preferred Network Time Protocol (NTP) server.

```
config routing
     config route inside-ntwk
      set destination network 192.168.0.0/16
      set gateway 192.168.1.1
 return
    return
   return
  return
  config ntp-client
   set server pool.ntp.org
return
  config cli
   set prompt nnos-e-vm>
  return
 return
return
```

The following section of the configuration contains all of the event log filters and targets.

```
config services
 config event-log
  config file eventlog
   set filter all error
  return
  config file access-log
   set filter access info
  return
  config file kernelsys
   set filter krnlsys debug
  return
  config file db
   set filter db debug
  return
  config file system
   set filter general info
   set filter system info
  return
  config file access
   set filter access info
  return
  config file dos
   set filter dosSip alert
  return
  config local-database
   set filter all error
  return
 return
return
```

The following section of the config provides some commonly used default system parameters; for more information on these properties, see the *Oracle Communications WebRTC Session Controller Media Engine Objects and Properties Reference Guide*.

```
config master-services
 config database
  set media enabled
 return
return

config vsp
 set admin enabled
 config default-session-config
  config media
   set anchor enabled
   config nat-traversal
    set symmetricRTP true
   return
   set rtp-stats enabled
  return
  config sip-directive
   set directive allow
  return
  config log-alert
  return
 return
 config tls
  config certificate sample
  return
 return
```

The following section of the configuration provides a sample policy rule to reject calls from a user with a URI that starts with 1000.

```
config policies
  config session-policies
   set default-policy vsp\policies\session-policies\policy default
   config policy default
    config rule sample-rule
     set description "sample rule to reject calls"
     config condition-list
      set from-uri-condition user match 1000
     return
     config session-config
      config sip-directive
       set directive refuse 400 "Please Pay Your Bill"
      return
     return
    return
   return
  return
 return
```

The folllowing configuration provides a sample dial-plan that takes a call with a Req URI domain of delegate.com and forwards it to the sample SIP gateway.

```
 config dial-plan
  config route sample-delegate
   set description "delegate to defined server"
   set peer server "vsp\enterprise\servers\sip-gateway sample-gateway"
   set request-uri-match domain-exact delegate.com
```

```
  return
 return
```

The following configuration provides a sample registration plan that takes a registration attempt with a domain of *xyz.com* and registers the endpoint locally. This is useful for cases where you want to register an endpoint locally for call testing purposes.

```
config registration-plan
  config route sample-accept-local
   set description "accept registers locally for this domain"
   set action accept
   set peer server "vsp\enterprise\servers\sip-gateway sample-gateway"
   set to-uri-match domain-exact xyz.com
  return
```

The following configuration provides a sample registration plan that takes a registration attempt with a domain of delegate.com and proxies the registration to the attached PBX or feature server.

```
config route sample-delegate
   set description "delegate to the defined server"
   set peer server "vsp\enterprise\servers\sip-gateway sample-gateway"
   set to-uri-match domain-exact delegate.com
  return
 return
```

The following configuration provides a sample SIP gateway that could be used for an attached PBX or feature server. You must edit the IP address to reflect the actual server IP or Fully Qualified Domain Name (FQDN).

```
config enterprise
 config servers
  config sip-gateway sample-gateway
   config server-pool
    config server sample-server
     set host 192.168.1.4
    return
   return
  return
 return

config external-services
return
config preferences
 config cms-preferences
 return
return
```

The following configuration provides two different sample permission sets. These permission sets modified and/or can be used with user accounts that you create.

```
config access
 config permissions super-user
  set cli advanced
 return
 config permissions view-only
  set cli disabled
  set ftp disabled
  set config view
```

```
  set actions disabled
  set templates disabled
  set web-services disabled
  set debug disabled
 return
return

config features
return
```

Oracle recommends that the storage-device fail-threshold be set to 200 MB.

```
services
storage-device
  fail-threshold 200 MB
```

# Enabling the ME Management System

Once you have configured an Ethernet interface, such as eth1, you can use your Web browser or native mobile application to point to the configured IP address of this interface to launch the ME Management System. The ME Management System provides a windows and menu user interface to configuring the ME.

# Bridging to Additional Ethernet Ports

Follow the steps in this section if you need to configure VMware on a Window platform to use two bridged networks. By default, VMware allows the following functionality:

- One bridged interface (to the first host network interface)
- One NAT interface
- One host-only interface

To create two bridged interfaces, you will need to

1. add an additional VMnet associated with a second interface, and
2. edit the VM configuration file to use the new VMnet.

## Adding an Additional VMnet

To add an additional VMnet, perform the following steps:

1. Halt all VMs currently running on this x86-based PC or server.
2. Launch the **vmnetcfg.exe** application from the VMware Player installation directory (c:\Program Files\VMware\VMware Player\vmnetcfg.exe).
3. Select the **Host Virtual Network Mapping** tab.
4. Select a VMnet to use for the second network interface card (NIC), such as VMnet3.
5. From the drop-down men, select the NIC you wish to connect to this VMnet.

If you want to have more control over which VMnet0 which connects to the first NIC perform the following steps:

1. Select the **Automatic Bridging** tab.

2. In the **Automatic Bridging** box, de-select the **Automatically choose and available physical network adapter to bridge to VMnet0**.
3. Select the **Host Virtual Network Mapping** tab.
4. Select a VMnet to use for the first NIC, such as VMnet2.
5. From the drop-down menu, select the NIC you wish to connect to this VMnet.

> **Note:** You can use VMnet0 to assign to a specific NIC. However, avoiding VMnet0 will indicate to a later user of the VMs configuration file that specific NICs were assigned to the VMs virtual interfaces, thus removing any questions about the automatic nature implied with VMnet0 on any particular system.

### Editing the VM Configuration File

You will need to edit the VMware configuration file to include the second NIC with the VMware Player. Perform the following steps:

1. Halt all VMs currently running on this x86-based PC or server.
2. Using Windows Explorer, open the Oracle ME folder.
3. Using a text editor such as Notepad, open the file **nnos-e-vm.vmx**.
4. At the bottom of the file add the following lines, substituting the desired VMnets for the Ethernet interfaces:
   - ethernet0.connectionType = "custom"
   - ethernet0.vnet = "vmnet0"
   - ethernet1.connectionType = "custom"
   - ethernet1.vnet = "vmnet3"
5. Ensure that there are no other lines in the file specifying **ethernet X .connectionType = "XXXXX"**.

## Media Engine Virtual Machine Troubleshooting

Oracle makes every effort to test the VM in a variety of customer environments. This section covers recently reported issues directly from ME VM customers. If you discover an issue with the VM that we need to know about, contact Oracle Customer Support for assistance.

# 15

# Upgrading WebRTC Session Controller Media Engine From an Earlier Version

This chapter provides instructions for upgrading to Oracle Communications WebRTC Session Controller Media Engine (ME) from an earlier version.

To upgrade the ME to 7.2, you must do the following:

- Backup your ME configuration, files, and databases
- Install Oracle Linux 7
- Install 7.2 from a USB stick
- Restore your configuration, files, and databases on the ME

## Backing Up the Media Engine Configuration, Files, and Databases

Prior to upgrading the ME, you must back up your configuration so that you can restore it once the 7.2 software is installed.

This upgrade repartitions and reformats the disk, so any items you need (such as .wav files and CDRs) need to be backed up.

**To backup the ME configuration:**

1. Insert your USB stick into the ME system to be upgraded.
2. Mount the USB stick using the mount usb command.

   ```
   NNOS-E>mount usb
   Device is mounted
   ```
3. Backup your configuration onto the USB stick using the **restore-stick-create config-backup** command.

   ```
   NNOS-E>restore-stick-create config-backup
   ```
   A folder named “setup”, containing the backed up configuration, is written to the USB stick.
4. Unmount the USB stick using the **unmount usb** command.

   ```
   NNOS-E>unmount usb
   ```

**To backup files and databases:**

1. Execute the **database-backup backup system** command.

   ```
   NNOS-E>database-backup backup system <databasePath>
   ```
2. Repeat this process for as many databases as you need to back up.

For more information on the **database-backup** command, see the *Oracle Communications WebRTC Session Controller Media Engine Objects and Properties Reference* guide.

## Installing Oracle Linux 7

Once you have backed up your configuration, you can install Oracle Linux 7.

**Note:** You must install Oracle Linux version 7.0 or higher.

For information on installing Oracle Linux, see <blue>Installing Oracle Linux 7.

## Installing the Media Engine

When Oracle Linux 7 is installed, you can download the ME installation file, copy it to a USB stick, and install the ME.

For more information on installing the ME, see <blue>Installing the Media Engine.

## Restoring Your Configuration, files, and Databases On the Media Engine

When the ME is installed, restore your configuration, files, and databases.

**To restore your ME configuration:**

1. When the ME has rebooted and the login prompt appears, insert the USB stick you created in <blue>Backing Up the Media Engine Configuration, Files, and Databases.
2. Mount the USB using the following command:

   ```
   NNOS-E>mount usb
   ```
3. Execute the **restart warm** command.

The configuration is loaded onto the ME.

**To restore your files and databases:**

1. Execute the **file fetch** command or SCP to copy any .wav files and archived CDRs to the ME.
2. Execute the **database-backup restore system** command to restore any previously backed up CDRs and database.
3. Repeat this process for as many databases as you need to restore.

For more information on the file fetch and database-backup commands, see the *Oracle Communications WebRTC Session Controller Media Engine Objects and Properties Reference* guide.

# 16

# Troubleshooting a WebRTC Session Controller Installation

This chapter describes how to troubleshoot Oracle Communications WebRTC Session Controller installations.

## Troubleshooting a Signaling Engine Installation

The WebRTC Session Controller Signaling Engine (Signaling Engine) installer and the Domain Configuration Wizard write information to log files. You can check those log files for information about errors and actions performed during the installation.

### Signaling Engine Installation Log Files

The Signaling Engine installation logs can be found at *Central_inventory_location*/**oraInventory/logs**, where *Central_inventory_location* is the directory path to the **oraInventory** directory. If you do not know the location of your Oracle Inventory directory, you can find it in the `oraInst.loc` file in the directory, **/etc/oraInst.loc**.

The following install log files are written to the log directory:

- `install`*`date-time-stamp`*`.log`

  This is the main log file.

- `install`*`date-time-stamp`*`.out`

  This log file contains the output and error streams during the installation.

- `installProfile`*`date-time-stamp`*`.log`

  This log file contains the overall statistics like time taken to complete the installation, as well as configuration, memory and CPU details.

- `oraInstall`*`date-time-stamp`*`.log`

  This log file contains the output stream of the copy session.

- `oraInstall`*`date-time-stamp`*`.err`

  This log file contains the error stream of the copy session.

#### Changing the Installer Logging Level

Use the `-logLevel` parameter from the command line when you start the installer. For example:

```
java -jar wsc_generic.jar -logLevel info
```

Valid value for `-logLevel` are listed below from most detailed to least detailed:

- `severe`
- `warning`
- `info`
- `config`

You can also specify logging details from detailed to least detailed in the following manner:

- `fine`
- `finer`
- `finest`

## Signaling Engine Domain Configuration Log Files

If you encounter errors when configuring a Signaling Engine domain, you can start the Fusion Middleware Configuration Wizard with the appropriate logging options.

To enable domain configuration logging, navigate to *Oracle_home*/**oracle_common/common/bin** and start `config.sh` with the `-log` and `-log_priority` options:

```
./config.sh -log=log_filename -log_priority=log_level
```

Table 16–1 describes the `-log` and `-log_priority` options.

***Table 16–1 Configuration Wizard Log File Options***

| Parameter | Description |
|---|---|
| `-log` | Specify the location of your log file.<br>If you specify a log file name, it is created in the same directory as the `config.sh` script unless you add a path component. Log files are otherwise created in *Oracle_home*/**logs**.<br>Other values that can be specified with `-log` are:<br>■ `stdout`<br>This writes the error message to the standard output stream.<br>■ `stderr`<br>This writes the error messages to the standard error stream.<br>■ `disable`<br>This disables default logging so that no log files are generated in *Oracle_home*/**logs**. |

*Table 16–1 (Cont.) Configuration Wizard Log File Options*

| Parameter | Description |
|---|---|
| -Log_priority | Specify the level of detail you want included in your logs.<br>The acceptable values are listed below:<br>■ OFF<br>■ SEVERE<br>■ WARNING<br>■ INFO<br>■ CONFIG<br>■ FINE<br>■ FINER<br>■ FINEST |

# Troubleshooting a Media Engine Installation

WebRTC Session Controller Media Engine (Media Engine) implements specialized troubleshooting commands that can be executed at the console command line interface (CLI).

For detailed troubleshooting instructions, see the discussion of Media Engine system monitoring in *Oracle Communications WebRTC Session Controller System Administrator's Guide*.

## Checking Media Engine Event Logs

You can use the following CLI command to display events logged to the Media Engine's local database:

```
show event-log
```

Information in the Media Engine event log can provide details on software errors, connectivity issues, and incorrectly configured objects and properties.

## Checking for Software Faults

You can use the following CLI command to display any software subsystem faults:

```
show faults
```

In conjunction with configuration files, traces and event logs, information for the show faults command can be used to troubleshoot system issues.

## Checking for Hardware Issues

You can check the status of a Media Engine installation's hardware sensors using the CLI command:

```
show sensor-events
```

Likely failure points are power supplies and cooling fans.

## Checking for Networking Issues

You can check Media Engine's network interface status using the CLI command:

```
show interfaces
```