Alcatel-Lucent

VIRTUALIZED SERVICE ROUTER | RELEASE 14.0.R1

INSTALLATION AND SETUP GUIDE

Alcatel·Lucent

Disclaimers

Alcatel-Lucent products are intended for commercial uses. Without the appropriate network design engineering, they must not be sold, licensed or otherwise distributed for use in any hazardous environments requiring fail-safe performance, such as in the operation of nuclear facilities, aircraft navigation or communication systems, air traffic control, direct life-support machines, or weapons systems, in which the failure of products could lead directly to death, personal injury, or severe physical or environmental damage. The customer hereby agrees that the use, sale, license or other distribution of the products for any such application without the prior written consent of Alcatel-Lucent, shall be at the customer's sole risk. The customer hereby agrees to defend and hold Alcatel-Lucent harmless from any claims for loss, cost, damage, expense or liability that may arise out of or in connection with the use, sale, license or other distribution of the products in such applications.

This document may contain information regarding the use and installation of non-Alcatel-Lucent products. Please note that this information is provided as a courtesy to assist you. While Alcatel-Lucent tries to ensure that this information accurately reflects information provided by the supplier, please refer to the materials provided with any non-Alcatel-Lucent product and contact the supplier for confirmation. Alcatel-Lucent assumes no responsibility or liability for incorrect or incomplete information provided about non-Alcatel-Lucent products.

However, this does not constitute a representation or warranty. The warranties provided for Alcatel-Lucent products, if any, are set forth in contractual documentation entered into by Alcatel-Lucent and its customers.

This document was originally written in English. If there is any conflict or inconsistency between the English version and any other version of a document, the English version shall prevail.

Table of Contents

List of Tables

List of Figures

Preface

About This Guide

This guide describes how to install and set up the Alcatel-Lucent Virtualized Service Router (VSR) portfolio of products.

Note: Unless otherwise noted, the information in this guide applies to all of the supported products in the VSR portfolio.

The following VSR products are currently supported:

- Virtualized Service Router - Simulator (VSR-SIM)
- Virtualized Service Router - Route Reflector (VSR-RR)

This guide includes:

- a description of the VSR system architecture
- a functional overview of the VSR
- requirements for the physical hardware and the virtual machine (VMs) supporting the VSR system
- procedures to configure and run the VSR

Audience

This guide is intended for network administrators who are responsible for configuring VSRs. It is assumed that the network administrators have an understanding of the following:

- x86 hardware architecture
- Linux system installation, configuration, and administration methods
- basic XML syntax
- 7750 SR and 7950 XRS chassis components

- 7750 SR OS CLI
- networking principles and configurations

List of Technical Publications

After the installation process of the required VSR product is completed, refer to the following SR OS documents for information about the software configuration and the command line interface (CLI) that is used to configure network parameters and services:

- 7750 SR OS Basic System Configuration Guide

  This guide describes basic system configurations and operations.
- 7750 SR OS System Management Guide

  This guide describes system security and access configurations as well as event logging and accounting logs.
- 7750 SR OS Interface Configuration Guide

  This guide describes card, Media Dependent Adapter (MDA), and port provisioning.
- 7750 SR OS Router Configuration Guide

  This guide describes logical IP routing interfaces and associated attributes such as IP addresses, ports, link aggregation groups (LAGs), as well as IP-based and MAC-based filtering, VRRP, and Cflowd.
- 7750 SR OS Routing Protocols Guide

  This guide provides an overview of routing concepts and provides configuration examples for RIP, OSPF, IS-IS, Multicast, BGP, and route policies.
- 7750 SR OS MPLS Guide

  This guide describes how to configure Multiprotocol Label Switching (MPLS) and Label Distribution Protocol (LDP).
- 7750 SR OS Services Guide

  This guide describes how to configure service parameters such as service distribution points (SDPs), customer information, and user services.
- 7750 SR OS OAM and Diagnostic Guide

  This guide describes how to configure features such as service mirroring and Operations, Administration and Management (OAM) tools.
- 7750 SR OS Triple Play Guide

  This guide describes Triple Play services and support provided by the 7750 SR and presents examples to configure and implement various protocols and services.
- 7750 SR OS Quality of Service Guide

  This guide describes how to configure Quality of Service (QoS) policy management.
- OS Multi-Service ISA Guide

  This guide describes services provided by integrated service adapters such as Application Assurance, IPSec, ad insertion (ADI) and Network Address Translation (NAT).

Searching for Information

You can use Adobe Reader, Release 6.0 or later, to search one or more PDF files for a term.

To search for specific information in this guide

From the Adobe Reader main menu, choose Edit > Search or Advanced Search. The Search panel opens.

1. From the Adobe Reader main menu, choose Edit > Search or Advanced Search. The Search panel opens.
2. Click on the In the current document radio button.
3. Enter the term to search for.
4. Select the following search criteria, if required.
   - Whole words only
   - Case-Sensitive
   - Include Bookmarks
   - Include Comments

You can expand the entries by clicking on the + symbol.

To search for specific information in multiple documents

Note: The PDF files that you search must be in the same folder.

1. From the Adobe Reader main menu, choose Edit > Search or Advanced Search. The Search panel opens.
2. Click on the All PDF Documents radio button.
3. Choose the folder in which to search using the drop-down menu.
4. Enter the term to search for.
5. Select the following search criteria, if required:
   - Whole words only
   - Case-Sensitive
   - Include Bookmarks
   - Include Comments
6. Click on the Search button. Adobe Reader displays the search results.

You can expand the entries for each file by clicking on the + symbol.

Technical Support

If you purchased a service agreement for your Alcatel-Lucent Virtualized Service Router (VSR) and related products from a distributor or authorized reseller, contact the technical support staff for that distributor or reseller for assistance. If you purchased an Alcatel-Lucent service agreement, follow this link to contact an Alcatel-Lucent support representative and to access product manuals and documentation updates:

http://support.alcatel-lucent.com

Report documentation errors, omissions and comments to:

Documentation.feedback@alcatel-lucent.com

Include document name, version, part number and page(s) affected.

1 — Getting Started

1.1 — In This Chapter

This chapter describes the process flow to install and configure the Alcatel-Lucent Virtualized Service Router (VSR).

1.2 — Alcatel-Lucent VSR Installation and Setup Process Overview

Table 1.1 lists the tasks necessary to install and setup the VSR.

The information in this guide is presented in the order of the tasks that you must perform.

Table 1.1 — Installation and Configuration Process

Area	Task	Chapter
Product information	Introduction to the Alcatel-Lucent VSR	Chapter 2, VSR Overview
Pre-Installation tasks	Server requirements	Chapter 3, Hypervisor Requirements
	License requirements	Chapter 4, VSR Software Licensing
	Mapping network interfaces	Chapter 5, VSR Network Connectivity
	VSR Initial Configuration and Setup	Chapter 6, Customizing Initial Configuration of the VSR
Installation tasks	VSR installation on a Libvirt Kernel-based Virtual Machine (KVM)	Chapter 8, Deploying the VSR VM on Linux Hosts Using the KVM/QEMU Hypervisor
	VSR installation on VMware ESXi	Chapter 7, Deploying the VSR VM on VMware ESXi Hosts

Table 1.1 — Installation and Configuration Process (Continued)

Area	Task	Chapter
Post-Installation tasks	Verifying and Troubleshooting the Installation	Chapter 9, Verifying and Troubleshooting the VSR Installation

2 — VSR Overview

2.1 — In This Chapter

This chapter introduces the Alcatel-Lucent Virtualized Service Router (VSR). The topics in this chapter include:

- VSR Overview
- VSR Deployment Models
- VSR Networking
- VSR Installation Overview
- VSR Software Packaging
- Supported VSR-SIM Features
- Supported VSR-RR Features

2.2 — VSR Overview

Network function virtualization (NFV) represents a new trend in networking where network functions that previously depended on custom hardware can now be deployed on commodity hardware using standard IT virtualization technologies for added flexibility. For network operators, the benefits of NFV include:

- reduced CAPEX by using industry-standard hardware that is potentially easier to upgrade
- reduced OPEX (space, power, cooling) by consolidation of multiple functions on fewer physical platforms
- faster and simpler testing and rollout of new services
- more flexibility to scale capacity up or down, as needed
- ability to move or add network functions to a location without necessarily needing new equipment

This guide describes the following NFV products that are part of the Alcatel-Lucent Virtualized Service Router (VSR) portfolio:

- Virtualized Service Router - Simulator (VSR-SIM)
- Virtualized Service Router - Route Reflector (VSR-RR)

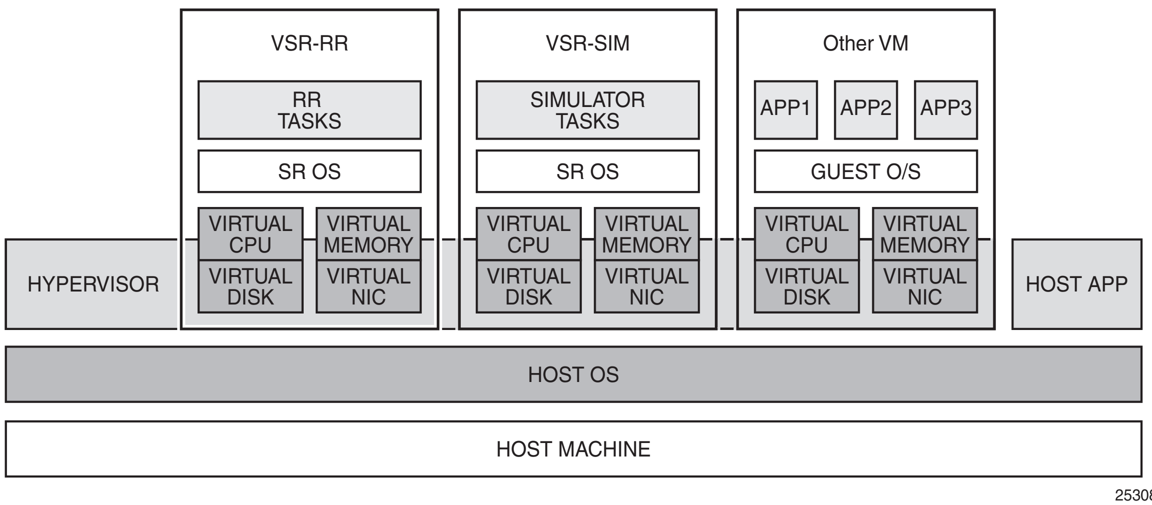
The VSR-SIM simulates the control, management, and forwarding functions of a 7750 SR or 7950 XRS router. The VSR-SIM runs the same Service Router operating system (SR OS) that runs on the 7750 SR and 7950 XRS hardware-based routers and has the same feature set and operational behavior as those platforms. Configuration of interfaces, network protocols, and services is done the same way as it is on a physical 7750 SR or 7950 XRS system.

The VSR-SIM is suitable for trials, training and education, and network simulation. It is optimized for testing and lab environments. It is not intended for deployment in a production network.

The VSR-RR virtualizes the function of a control-plane BGP route reflector. It runs the Service Router operating system (SR OS) and is functionally and operationally equivalent to an SR-series router deployed as control-plane BGP route reflector. The VSR-RR is fully supported for live deployments, but it is not designed to support transit traffic forwarding.

Figure 2.1 illustrates the general concept of a VSR. Hypervisor software running on the host machine creates and manages virtual machines (VMs), each of which is an abstraction of a physical machine with some subset of the CPU, memory, and other resources of the system. A VSR-RR or VSR-SIM is created by running an x86-optimized version of the SR OS software in one or more of these virtual machines. In virtualization terms, the SR OS is a guest operating system of a virtual machine. VSR virtual machines can be deployed in combination with other virtual machines on the same server, including virtual machines that run guest operating systems other than the SR OS.

Figure 2.1 — VSR Concept



The host machine supporting a VSR-RR or VSR-SIM is a common-off-the-shelf (COTS) x86-based server that must be sourced directly from the vendor or reseller of such a system. The minimum requirements of the host server are described in Chapter 3, Hypervisor Requirements.

The host machine must run a hypervisor that is compatible with the VSR-RR and VSR-SIM software. The supported hypervisors for the VSR-RR and VSR-SIM 13.0 software release are:

- KVM/QEMU installed on Centos 6.5, Centos 7.0, or Red Hat Enterprise Linux 7.0
- VMware ESXi 5.5

Note: Alcatel-Lucent does not resell or package hypervisor software. You must acquire the hypervisor software from a vendor or open-source repository of such software.

2.2.1 — VSR Key Features

The key features of the VSR are described in Table 2.1.

Table 2.1 — VSR Features

Features	Description
Symmetric Multiprocessing (SMP)	The VSR inherits the advanced Symmetric Multiprocessing support in the base SR OS software. With SMP, the application software tasks are scheduled to all available CPU cores so that they can run concurrently, resulting in faster performance in multi-core systems.
Support for different networking options	VSR software includes drivers for E1000 and VirtIO NIC interfaces. VirtIO is a para-virtualization technology where the guest and the hypervisor cooperate to provide overall better performance than complete device emulation by the hypervisor. The virtual NIC interfaces of a VSR can be connected to a virtual switch in the host or they can be passed through directly to a physical NIC port, allowing for significant deployment flexibility.
Support for different hypervisors	The VSR can run on virtual machines managed and created by one of the following hypervisors: • KVM/QEMU • VMware ESXi
Flexible host machine requirements	The requirements for the host machine are minimal. A VSR-RR or VSR-SIM requires the host machine to have an Intel CPU with a Sandy Bridge (or later) micro-architecture. A minimum of 8 GB of memory is needed. A wide range of physical NICs can be accommodated.
Management by 5620 SAM	The 5620 can manage the VSR in the same way as the SR-series routers.

2.3 — VSR Deployment Models

The VSR can be deployed as one of two models: integrated or distributed. The VSR-SIM supports both models, depending on the chassis type that is emulated; the VSR-RR supports only the integrated model.

2.3.1 — Integrated Model

The integrated VSR model uses a single VM to represent a network element. All control plane, management, and forwarding functions are performed by one virtual machine.

Note: The integrated model is always used for VSR-RR deployments. It is also supported for a VSR-SIM when the configured chassis type is SR-C4 or SR-C12.

You can increase the scale of an integrated system by adding virtual CPUs and virtual memory to the VM; however, such changes require the VM to be shut down and restarted.

Note: Port Cross-Connect (PXC) is not supported on integrated vSIMs, where CPM and forwarding plane run on the same host machine.

2.3.2 — Distributed Model (VSR-SIM Only)

The distributed model uses two or more virtual machines to represent the network element. One or two of the virtual machines execute the functions of CPM cards in physical systems and another set of one or more virtual machines execute the functions of line cards in physical systems (IOMs/IMMs in a 7750 SR chassis or XCMs in a 7950 XRS chassis). The CPM and IOM VMs of a VSR-SIM must be able to communicate with each other through physical or virtual switches.

The distributed VSR model is supported for VSR-SIMs that emulate one of the following chassis types:

- 7750 SR-a4
- 7750 SR-a8
- 7750 SR-7
- 7750 SR-12
- 7750 SR-12e

- 7950 XRS-16c
- 7950 XRS-20

The distributed VSR-SIM is suitable for simulating specific features of a hardware-based router. For example, a distributed VSR-SIM can simulate and validate features such as HA and ISSU that cannot be tested on an integrated VSR-SIM. Because a distributed VSR-SIM must simulate each of the CPMs and IOMs, on individual VMs, it may consume more physical server resources than an integrated VSR-SIM.

2.3.2.1 — Examples of Distributed VSR Deployment

The examples in this section show how a VSR-SIM can be configured using standard x86 servers. Distributed VSR-SIMs can be deployed using one or more servers. Figure 2.2 shows an example of a virtualized 7750 SR-7 that uses four VMs that run on one physical server.

Figure 2.2 — Distributed System on a Single Server

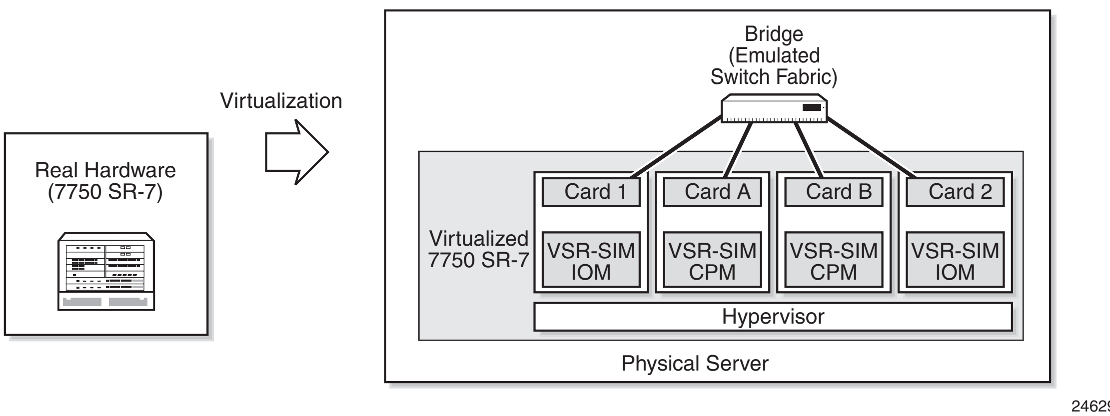
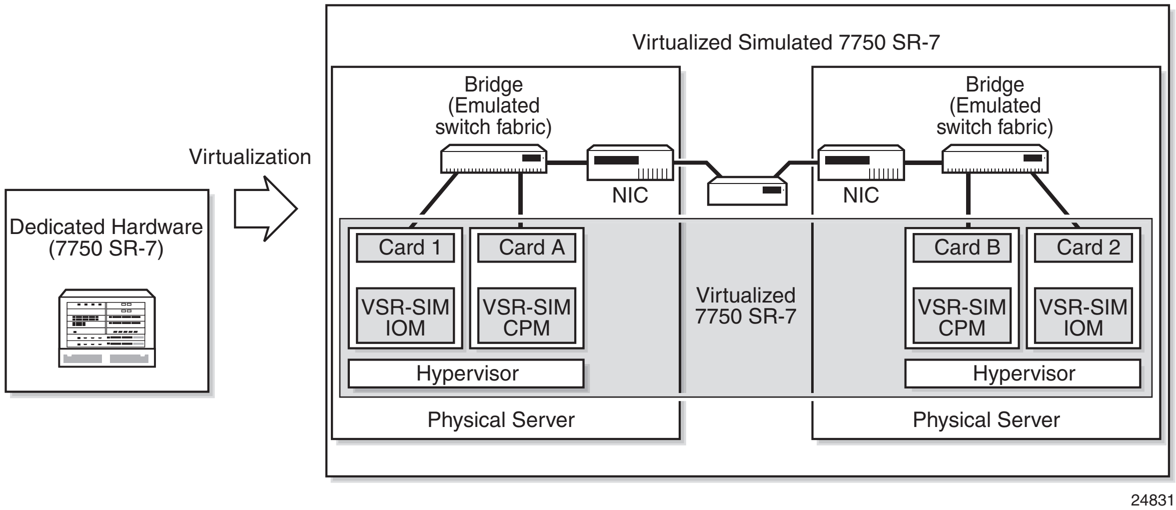


Figure 2.3 shows an example of a virtualized 7750 SR-7 that runs on two physical servers. Two VMs run on each physical server. Each server runs a control VM, and when both are operational, one VM will operate as the active control VM and the other will operate as the standby control VM. Each host connects one of their physical NIC ports to an internal virtual switch (vSwitch) and this allows communication between the CPM and IOM VMs of the VSR-SIM.

Figure 2.3 — Distributed System on Multiple Servers



2.4 — VSR Networking

A virtual machine used by an Alcatel-Lucent VSR can have one or more virtual NIC ports. In the networking models supported by the VSR-SIM and VSR-RR, each virtual NIC port is internally connected to a logical interface within the host. The logical host interface may map directly to a physical NIC port/VLAN or it may connect to a vSwitch within the host. The supported vSwitch options depend on the hypervisor that is running.

- If the host is running the KVM hypervisor, the vSwitch can be a standard Linux Bridge (installed in Centos/RedHat with bridge-utils) or Open vSwitch. For more information about Open vSwitch, refer to www.openvswitch.org.
- If the host is running the VMware ESXi hypervisor, the vSwitch can be a vSphere standard switch or a vSphere distributed switch.

If a vNIC port is connected to a vSwitch, a physical NIC port/VLAN must be added as a bridge port of the vSwitch in order for traffic sourced by or destined for the vNIC interface to reach physical networks connecting the host to other hosts.

For detailed information about the vNIC interfaces and their use in the SR OS guest, see Chapter 5, VSR Network Connectivity.

2.5 — VSR Installation Overview

The following steps show the general workflow to install a VSR virtual machine on a host machine:

1. Set up and install the host machine, including the host operating system.
2. Install the hypervisor software on the host machine.
3. Configure the host networking.
4. Download the VSR software as a QCOW2 disk image or OVA archive.
5. Obtain the license key file.
6. Define the properties of the virtual machine.
7. Start the virtual machine.

2.6 — VSR Software Packaging

The VSR software is available for download from OLCS in a ZIP file format with a name such as TiMOS-SR-13.0.Rn-vm.zip. The VSR software images are stored in virtual disk images in the ZIP file.

The **sros-vm.qcow2** file stored in the \vm\7xxx-i386\ directory is a QCOW2 disk image that contains the VSR software; this is typically used with KVM/QEMU virtual machines.

The **sros-vm.ova** file stored in the \vm\7xxx-i386\ directory is an OVA archive file that contains a VMDK disk image. The contents of the VMDK disk image are the same as the QCOW2 disk image, but the file format is different.

The OVA file is convenient for VMware deployments because vCenter can completely instantiate a VM from the OVA file. For more information on this topic, see Chapter 7.

The QCOW2 or VMDK disk image (visible when the file is attached as a disk device of a VSR VM) contains files that are necessary to boot and run a VSR, including:

- bootloader files
- image binaries for the VSR
- a generic BOF file

2.7 — Supported VSR-SIM Features

The VSR-SIM software supports most features and functions of the 7750 SR or 7950 XRS router that is being emulated, including:

- IPv4 and IPv6 unicast and multicast forwarding and dependencies on protocols such as ARP, IPv6 neighbor discovery, ICMPv4, ICMPv6, IGMP, and PIM
- Layer 3 routing protocols such as BGP, OSPF, and IS-IS
- MPLS features such as LDP, RSVP, DS-TE, LDP-over-RSVP, BGP-3107, 6PE and MPLS shortcuts
- OAM features such as BFD, IP ping and traceroute, LSP ping and traceroute, Ethernet OAM (CFM and PM), and TWAMP
- SAP, network, and subscriber interfaces with relevant statistics
- MAC and IP filters (ACLs)
- Layer 2 and Layer 3 services including VLL, VPLS, VPRN, and MVPN, along with inter-AS support
- QoS features including classification, marking, and policing
- management features including Telnet, SSH, SNMP, and Netconf

Note: The VSR-SIM has limited packet forwarding performance compared to a VSR-RR or a physical 7750 SR or 7950 XRS. A maximum of 250 pps can be received and forwarded by each port (vNIC) of the VSR-SIM.

2.8 — Supported VSR-RR Features

The VSR-RR software supports all the key features of a 7750 SR or 7950 XRS router that are relevant to the operation of a control-plane BGP route reflector, including:

- handling of locally originating and terminating IPv4 and IPv6 control packets, including dependencies on protocols such as ARP, IPv6 neighbor discovery, ICMPv4, and ICMPv6
- IGP protocols, including OSPFv2, OSPFv3, and IS-IS
- complete BGP functionality, including support for all address families, add-paths, best path selection controls, peer tracking, next-hop-resolution policies, import and export policies, graceful restart, rapid-update, rapid-withdrawal, RT constraint, ORF, and path MTU discovery

Note: The VSR-RR is not designed to handle transit traffic forwarding; you should avoid configurations that set the VSR-RR as the BGP next hop.

Alcatel-Lucent recommends that you disable the installation of all BGP routes in the route table to minimize the resources consumed by the VSR-RR.

3 — Hypervisor Requirements

3.1 — In This Chapter

This chapter describes the requirements for a VSR VM host including hardware specifications of the host machine, its operating system, and the hypervisor software installed to support virtualization.

Note: Alcatel-Lucent does not provide or sell third-party server hardware or software for the host machine. You must acquire the components described in this chapter and install them according to vendor instructions.

The topics in this chapter include:

- Host Machine Hardware Requirements
- Host Machine Hypervisor Software

3.2 — Host Machine Hardware Requirements

The minimum hardware requirements for a host machine to support the VSR are:

- 1 Intel x86 CPU with a minimum of 4 cores, using the Sandy Bridge micro-architecture or later (for example, Ivy Bridge or Haswell)
- 8 GB of DRAM memory
- NIC card with GE or 10 GigE interfaces

A host machine configured to meet the minimum requirements can support a single VSR virtual machine and no other VMs.

The total requirements of the system must be determined to configure a host machine to support multiple VSR VMs, In addition to the host's CPU and memory requirements, you will also need to calculate and add together the CPU and memory requirements of each VM.

```
total system requirements = (sum of N VM's CPU and memory) +
(sum of host machine CPU and memory)
```

Note: Care must be taken to ensure that the host machine's CPU and memory resources are not oversubscribed, especially when one or more of the VMs are part of a VSR system.

Each VSR VM requires a minimum of one virtual CPU and a minimum of 4 GB of VM memory; these resources must be reserved for the VM. The performance and stability of the VSR system may be negatively impacted if these resources are not reserved and the CPU or memory resources are over-subscribed.

Note: Alcatel-Lucent recommends that a minimum of two virtual CPUs should be allocated to a VSR to ensure that the VSR VM can operate in the 64-bit mode. Additional vCPU may also be necessary to meet the performance requirements of the VSR system, especially when functioning as a VSR-RR. Additional memory may also be required to meet the scaling objectives of the VSR system.

With one N-core CPU, the total (non-oversubscribed) number of virtual CPUs available to all the virtual machines is calculated as follows:

```
number of virtual CPUs available = 2*N - 1
```

This calculation assumes that the hyper-threading feature of the Intel CPU is enabled in the BIOS configuration, and a minimum of one virtual CPU is needed for proper operation of the host O/S and hypervisor software.

With N GB of DRAM in the host machine, the total (non-oversubscribed) amount of memory available to all the virtual machines is calculated as:

```
total (non-oversubscribed) amount of memory available = N-4
```

This calculation assumes that a minimum 4GB of memory is needed for proper operation of the host OS and hypervisor software.

3.3 — Host Machine Hypervisor Software

The hypervisor or virtual machine monitor (VMM) software runs on the host machine to create and manage the VMs, and provide the guest OS in each VM with an abstraction of the physical machine. SR OS Release 13.0 supports the following hypervisor options for the VSR VMs:

- VMware ESXi

  See Chapter 7, Deploying the VSR VM on VMware ESXi Hosts for detailed installation information.
- Libvirt Kernel-based Virtual Machine (KVM) /QEMU

  See Chapter 8, Deploying the VSR VM on Linux Hosts Using the KVM/QEMU Hypervisor for detailed installation information.

3.3.1 — VMware ESXi Support

You can install the VSR virtual machines on hosts running the VMware ESXi hypervisor. Only ESXi version 5.5 is supported with the SR OS 13.0 software.

Note: Alcatel-Lucent recommends deployment of the vSphere vCenter server and use of the vSphere Web Client GUI for managing the virtual machines in a VMware environment.

The following VMware features are supported with VSR virtual machines:

- e1000 vNIC interfaces
- vNIC association with a vSphere standard switch
- vNIC association with a vSphere distributed switch
- vMotion
- High Availability

Non-supported features include VMXNET device adapter support, DRS, fault tolerance, and Storage vMotion

3.3.2 — KVM/QEMU Support

You can install the VSR virtual machines on hosts running the Kernel-Based Virtual Machine (KVM) hypervisor. KVM is supported on host machines running a Linux operating system and relies on Quick Emulator (QEMU) to provide a complete hypervisor solution. KVM and QEMU are open source software.

KVM is a Linux kernel module that allows QEMU to take advantage of the hardware virtualization features of the CPU (for example, VT-x on Intel CPUs). QEMU is a user space program that emulates a broad range of devices including CPUs, disks, PCI express chipsets, USB devices, and serial ports.

QEMU version 1.5.3-60 or later is recommended for use with VSR virtual machines running SR OS version 13.0 software.

Alcatel-Lucent recommends use of the Libvirt software package to manage the deployment of VMs in a KVM/QEMU environment. Libvirt is an open source software that provides a set of APIs for creating and managing the virtual machines on a host machine, independent of the hypervisor. Libvirt uses XML files to define the properties of virtual machines and virtual networks. It also provides a convenient virsh command line tool.

Libvirt version 1.1.1-29 or later is recommended for use with VSR virtual machines running SR OS version 13.0 software.

3.3.2.1 — Linux O/S Support

The KVM/QEMU hypervisor requires a Linux operating system. While many Linux distributions and versions support KVM and QEMU, only the following combinations are fully tested and supported with VSR virtual machines running SR OS 13.0 software:

- Centos 6.5
- Centos 7.0 (with 3.10.0-123 kernel)
- Red Hat Enterprise Linux 7.0 (with 3.10.0-123 kernel)

4 — VSR Software Licensing

4.1 — In This Chapter

This chapter describes how software license keys are applied to the VSR virtual machines (VMs) to authorize software use for the VSR-RR and VSR-SIM functions. The topics in this chapter include:

- VSR-RR License Keys
- VSR-SIM License Keys
- Checking the License Status

4.2 — VSR-RR License Keys

When you purchase software licenses for the VSR-RR, your Alcatel-Lucent account representative will provide you with corresponding VSR-RR license key files. If multiple VSR-RRs are part of the same order. you may receive one file for each VSR-RR, or you may receive one file that contains the license keys for all the VSR-RRs.

Each VSR-RR must be linked to a license file that contains its license key. The **license-file** boot option (BOF) parameter makes this association on the VSR-RR. The **license-file** parameter can by specified by editing the BOF file (before or after boot-up), or by including it in the configuration data of the VM and passing it to the guest (SR OS) as SMBIOS information. The **license-file** parameter can reference a file stored on a local disk (for example CF3:) or a file stored on an FTP server. See Chapter 6 for more information about SMBIOS parameters.

When the SR OS software starts booting and determines that the VM is a VSR-RR, it attempts to read and parse the associated license file that contains its license key. If a matching license key is not found, the SR OS allows the system to complete its boot-up procedures but causes a forced reboot to occur after 60 minutes.

A matching license key must meet the following criteria:

- the UUID of the VM matches the one encoded in the license key
- the VSR software version (the major release number) matches the one encoded in the license key
- the current date is greater than the encoded origination and start dates in the license key, and less than the encoded end date in the license key, if applicable

When there are multiple matching license keys in the license file, the most recent one is selected. If the selected license file is found to be corrupt, the situation is handled the same way as a missing license key and the 60 minute forced reboot timer is started.

4.3 — VSR-SIM License Keys

When you purchase software licenses for the VSR-SIM, your Alcatel-Lucent account representative will provide you with corresponding VSR-SIM license key files. If multiple VSR-SIMs are part of the same order. you may receive one file for each VSR-SIM, or you may receive one file that contains the license keys for all the VSR-SIMs.

Each VSR-SIM must be linked to a license file that contains a license key for each of its CPM VMs. If a VSR-SIM has two redundant CPM VMs, the license file for the VSR-SIM must contain at least two license keys: one for the active CPM and one for the standby CPM.

The **license-file** boot-option parameter of the VSR-SIM indicates the location of the license file, which can be a local disk location or an FTP server location. The **license-file** parameter can be specified by editing the BOF file (before or after boot-up), or by including it in the configuration data of the VM and passing it to the guest (SR OS) as SMBIOS information. See Chapter 6 for more information about the SMBIOS parameters.

When the SR OS software starts booting in a virtual machine and it determines (through SMBIOS information) that it is a CPM VM of a VSR-SIM, it attempts to read and parse the license file that contains its license key. If either CPM VM of a redundant system (or the single CPM VM of a non-redundant system) cannot find a matching license key, then the system is allowed to complete its boot-up procedures but a forced reboot occurs after 60 minutes.

A matching license key must meet the following criteria:

- the encoded product type in the license key is VSR-SIM (and not VSR-RR)
- the UUID of the virtual machine matches the one encoded in the license key
- the VSR software version (the major release number) matches the one encoded in the license key
- the current date is greater than the encoded origination and start dates in the license key, and less than the encoded end date in the license key, if applicable

When there are multiple matching license keys in the license file, the most recent one is selected. If the selected license file is found to be corrupt, the situation is handled the same way as a missing license key and the 60 minute forced reboot timer is started.

Note: The IOM VMs of a VSR-SIM do not need their own license keys; they inherit the license state of the system, as determined by the CPM VMs. The IOM VMs reboot immediately if the license state of the system is invalid.

4.4 — Checking the License Status

Once the VSR is operational, you can check the license status of the system. At the prompt, type the following:

show system license ↵

The following is sample output for a VSR-SIM with two CPM VMs; both CPM VMs in have a valid license:

```
A:vsr-sim# show system license

===============================================================================
System  License
===============================================================================
License status : monitoring, valid license record
Time remaining : 178 days 5 hours
===============================================================================

===============================================================================
Active  License [CPM A]
===============================================================================
License status : monitoring, valid license record
Time remaining : 178 days 5 hours
-------------------------------------------------------------------------------
License name   : network-operator.com
License uuid   : 8664885f-db82-4f0f-8f5f-0ba15275ae91
Machine uuid   : 8664885f-db82-4f0f-8f5f-0ba15275ae91
License desc   : joe@network-operator.com
License prod   : Virtual-SIM
License sros   : TiMOS-[BC]-13.0.*
Current date   : FRI SEP 04 18:55:16 UTC 2015
Issue   date   : MON AUG 24 22:48:41 UTC 2015
Start   date   : TUE SEP 01 00:00:00 UTC 2015
End     date   : TUE MAR 01 00:00:00 UTC 2016
===============================================================================
===============================================================================
Standby License [CPM B]
===============================================================================
License status : monitoring, valid license record
Time remaining : 178 days 5 hours
-------------------------------------------------------------------------------
License name   : network-operator.com
License uuid   : 6ef11a9c-d8ad-4912-8842-069c19cc37a7
Machine uuid   : 6ef11a9c-d8ad-4912-8842-069c19cc37a7
License desc   : joe@network-operator.com
License prod   : Virtual-SIM
License sros   : TiMOS-[BC]-13.0.*
Current date   : FRI SEP 04 18:55:17 UTC 2015
Issue   date   : MON AUG 24 22:48:41 UTC 2015
Start   date   : TUE SEP 01 00:00:00 UTC 2015
End     date   : TUE MAR 01 00:00:00 UTC 2016
===============================================================================
```

5 — VSR Network Connectivity

5.1 — In This Chapter

This chapter describes the network connectivity requirements of the Alcatel-Lucent VSR, including information about the relationship between the VSR ports, virtual NIC interfaces, and physical NIC interfaces.

The topics in this chapter include:

- Introduction to I/O Virtualization
- Mapping VSR-RR Ports to Virtual NIC Interfaces
- Mapping VSR-SIM Ports to Virtual NIC Interfaces
- Linux Host Networking

5.2 — Introduction to I/O Virtualization

A virtual machine (VM) can have one or more virtual NIC interfaces. The guest OS running in the virtual machine must have drivers that support each type of virtual NIC. The common types of virtual NICs in NFV applications are E1000 and VirtIO.

The E1000 is a generic driver for legacy Intel NICs. When a virtual NIC interface is configured to use an E1000 driver, the hypervisor (QEMU or VMware ESXi) provides complete software emulation of the physical NIC so that the guest is unaware of the virtualization.

When a virtual NIC interface uses a VirtIO driver, the guest and the hypervisor coordinate to streamline the movement of data to and from the guest. In this case, the guest is aware of the virtualization; VirtIO is also called a paravirtualized driver.

5.2.1 — Choosing a Driver

The VSR software includes both the E1000 and VirtIO drivers. The E1000 driver must be used for the VMware ESXi hypervisor. For the KVM/QEMU hypervisor, the virtual NIC interfaces of the VSR can be configured to use either E1000 or VirtIO. Alcatel-Lucent recommends the use of the KVM VirtIO driver for better performance.

Note: VMXNET drivers (the VMware equivalent of the KVM/QEMU VirtIO drivers) are not supported in the current software release of the VSR.

5.2.2 — Supported Networking Models

Each E1000 and VirtIO virtual NIC interface of a virtual machine has a corresponding logical interface in the host software layer.

When KVM/QEMU is the chosen hypervisor, the logical interface in the host is often a TAP or MACVTAP interface.

When the virtual NIC interface is defined as a bridge port to a Linux bridge, a TAP logical interface is configured. A Linux Bridge is a software implementation of an Ethernet bridge with control running in the Linux user space (brctl) and forwarding done in the Linux kernel.

Each packet transmitted by the guest on the virtual NIC interface is forwarded according to the MAC forwarding table of the Linux Bridge. If a physical NIC port of the host is added as another port of the Linux Bridge, the guest can communicate with the external networks using the virtual NIC interface.

When the virtual NIC interface is defined for a direct pass-through mode, the logical interface in the host is a MACVTAP interface. Each packet transmitted by the guest on this virtual NIC interface is transmitted directly out the physical NIC port that is mapped one-to-one to the virtual NIC port.

When VMware ESXi is the chosen hypervisor, the same options are available with slight differences in implementation:

- a virtual NIC interface can be connected to a vSphere standard or distributed switch (a software implementation of an Ethernet bridge in the ESXi host)
- a virtual NIC interface can be mapped directly to a physical NIC port

The VSR virtual NIC interfaces support the bridge and direct pass-through models with both KVM/QEMU and VMware hypervisors.

The option of configuring a virtual NIC interface as a logical interface to a software-based Linux Bridge in the host machine is also supported using Open vSwitch (OVS) as a substitute for the Linux Bridge.

The OVS is an open-source software implementation of a multi-layer switch that supports standard bridging protocols, monitoring protocols (sFlow, Netflow), and programmatic extensions (Openflow, OVSDB). Its main components are a userspace daemon (ovs-vswitchd), a database daemon (ovsdb-server), and a kernel module. The kernel module implements the fast path using a flow cache table that is populated by the **ovs-vswitchd** control application.

The OVS model has several advantages over the Linux Bridges:

- Faster performance—the performance is approximately 50% better than the Linux Bridge
- Superior networking flexibility—support is provided for data-center overlay encapsulations, such as VxLAN and GRE

The decision to use the bridge model for a virtual NIC interface over the direct pass-through model should be based on the following considerations:

1. The bridge model allows virtual machines in the same host to exchange packets directly with each other, without leaving the host.
2. The bridge model has a lower throughput than the direct pass-through model, if the software bridge is a Linux Bridge. If the software bridge is OVS, then the performance is comparable.

3. The bridge model allows traffic associated with multiple virtual NIC interfaces (and multiple guests) to share the same physical NIC port.

   In a direct pass-through model, sharing a physical NIC port is only possible by associating each virtual NIC interface with its own 802.1Q VLAN tag on the shared physical NIC port.
4. When implemented with OVS, the bridge model supports more DC overlay encapsulation options, such as VxLAN and GRE.
5. The bridge model may mistakenly terminate some Ethernet control or management packets in the host rather than passing them through to the guest, as intended.

5.3 — Mapping VSR-RR Ports to Virtual NIC Interfaces

A VSR-RR provides several ports for network connectivity:

- 1 out-of-band management port that is designated A/1 in the SR OS (guest) configuration

  Port A/1 of the VSR-RR is always associated with the first virtual NIC interface of the virtual machine
- Up to 5 I/O ports for exchanging control and data packets with other physical or virtualized routers. The I/O ports belong to a logical 5-port MDA in the SR OS configuration and are therefore, designated by the identifiers 1/1/1, 1/1/2, 1/1/3, 1/1/4 and 1/1/5.

  Ports 1/1/1 through 1/1/5 of the VSR-RR are associated with the second through sixth virtual NIC interfaces, in order of port numbering

As described in Section 5.2, the choice of the type of driver and networking model (bridge or pass-through) is a per virtual NIC decision and, therefore a per port decision in the SR OS configuration.

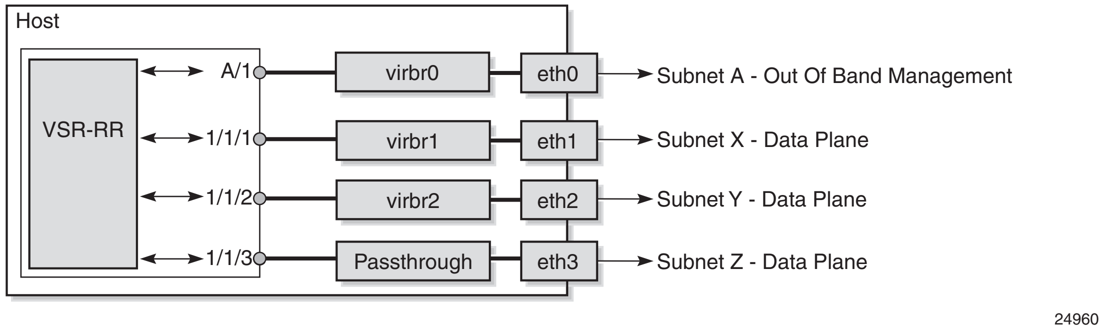
Note: Alcatel-Lucent recommends the use of the bridge model for the management port so that all VSR-RRs can be connected to a common management subnet.

5.3.1 — VSR-RR Port Mapping Examples

Figure 5.1shows an example of a potential mapping between the ports of a VSR-RR and the physical NIC ports of its host machine.

In this example, the virtual NIC interface corresponding to port A/1 is connected to a Linux Bridge called virbr0; this virtual bridge also has the physical NIC port eth0 as a bridge port. The virtual NIC interfaces corresponding to the I/O ports 1/1/1 through 1/1/3 are configured in direct pass-through mode and map one-to-one to the physical NIC ports eth1, eth2, and eth3 respectively.

Figure 5.1 — VSR-RR Port Mapping Example



5.4 — Mapping VSR-SIM Ports to Virtual NIC Interfaces

A VSR-SIM provides several ports for network connectivity. The exact set of ports and their mapping to virtual NIC interfaces depends on the type of chassis emulated by the VSR-SIM and the role of each VM in the context of the overall system.

A single VM of an integrated model system has the following ports:

- One out of band management port that is designated A/1

  The management port A/1 is associated with the first virtual NIC interface of the virtual machine
- One internal fabric port; the port is not explicitly visible in the configuration and show commands of the VSR-SIM

  The internal fabric port is associated with the second virtual NIC interface
- Up to 8 active I/O ports to exchange traffic with other physical or virtualized routers

  The 8 I/O ports belong to one or more MDAs, or their equivalents in different chassis types. Ports 1/1/1 and higher are associated with the third through tenth virtual NIC interfaces, in order of the port numbering

Note: The MDAs configured in the integrated VSR-SIM system may provide more than 8 ports in total. This is an acceptable configuration but only the first 8 ports (by numbering) can become operationally UP.

Each CPM VM of a distributed model system has the following ports:

- One out-of-band management port designated A/1 (or B/1, depending on the CPM)

  The management port is associated with the first virtual NIC interface of the virtual machine. The management port is exposed in the configuration and operational commands of the SR OS guest
- One internal fabric port that is associated with the second virtual NIC interface of the virtual machine

  The internal fabric port is not exposed in the configuration and operational commands of the SR OS guest

Each IOM VM of a distributed model system has the following ports:

Note: The terms MDA and IOM are used generically in the following descriptions and can have other representations depending on the chassis type. For example, in a 7950 XRS system the IOM is an XCM and the MDA is an XMA.

- One management port that is not exposed in the configuration and operational commands of the VSR-SIM

  The management port is associated with the first virtual NIC interface of the virtual machine

- One internal fabric port that is not exposed in the configuration and operational commands of the VSR-SIM

  The internal fabric port is associated with the second virtual NIC interface
- Up to 8 active I/O ports for exchanging traffic with other physical or virtualized routers

  The 8 I/O ports belong to one or more MDAs that are logically installed in the IOM slot. Ports n/1/1 and higher are associated with the third through tenth virtual NIC interfaces, in order of the port numbering

Note: The MDAs of the IOM may provide more than 8 ports in total. This is an acceptable configuration but only the first 8 ports (by numbering) can become operationally UP.

5.4.1 — Distributed VSR-SIM Switch Fabric

The second virtual NIC interface of every CPM and IOM virtual machine in a distributed model VSR-SIM is an internal fabric port, as described in Mapping VSR-SIM Ports to Virtual NIC Interfaces. This internal fabric port is used for all inter-card communications within a system.

When a CPM VM or IOM VM has to send control or data traffic to another VM (or multiple VMs) in the system, it transmits the messages using its internal fabric port and expects the messages to reach their destinations with layer 2 forwarding. Each distributed model VSR-SIM requires its own layer 2 tenant network in the data center. Each constituent VM connects its second vNIC interface to this tenant network. The bridge designated as "virbr1" in Figure 5.2 is a Linux bridge that provides fabric connectivity for the VSR-SIM shown in this example.

If the VSR-SIM is a distributed system on two or more servers, a vNIC interface of the VM is connected to a virtual bridge in the host (designated "virbr1" in Figure 5.2). The virtual bridge allows different VMs in the same host to be attached to the same subnet and use a single physical NIC port for communication external to the host. To support this model, one of the physical network ports on the server (for example, "eth1") is assigned to the bridge to provide connectivity between servers

5.4.2 — VSR-SIM Port Mapping Examples

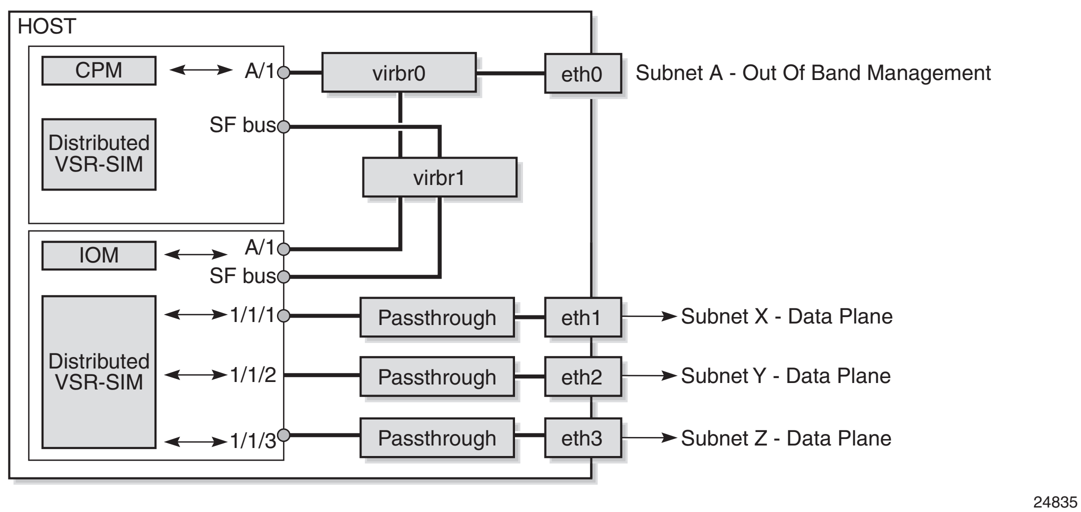
Figure 5.2 shows a networking example for a distributed model VSR-SIM with one CPM VM and one IOM VM that are both instantiated on the same host machine.

In this example, the virtual NIC interfaces corresponding to the management ports are connected to a common Linux Bridge called virbr0 that also has the physical NIC port eth0 as a bridge port.

The virtual NIC interfaces corresponding to the internal fabric ports are connected to another common Linux bridge called virbr1, but in this case no physical NIC ports are required.

The virtual NIC interfaces corresponding to the I/O ports 1/1/1 through 1/1/3 are configured in direct pass-through mode and map one-to-one to the physical NIC ports eth1, eth2 and eth3 respectively.

Figure 5.2 — Distributed VSR-SIM Port Mapping Example



5.5 — Linux Host Networking

This section provides brief examples of Linux Bridge configuration in a host machine running Centos 7.0 or Red Hat Enterprise Linux 7.0. For detailed instructions, please refer to the official documentation for the Linux distribution.

5.5.1 — Configuring a Linux Bridge

This example configuration describes how you can create a virtual bridge (called “virbr0”) and add port eth0 to the bridge.

The virbr0 interface has the static IP address that is used to access the host in this example. DHCP is disabled on this and all other interfaces.

1. Log in to the host.
2. Install the bridge-utils package, if required.
3. Create the Linux bridge:

   brctl addbr virbr0 ↵
4. Validate the bridge creation:

   brctl show ↵
5. Configure the Linux bridge interface:

   i. Open the interface configuration file for the virbr0 port in the vi editor (or other similar editor):

      vi /etc/sysconfig/network-scripts/ifcfg-virbr0 ↵

   ii. Configure the entries as follows:

      ```
      DEVICE=virbr0
      TYPE=Bridge
      BOOTPROTO=none
      ONBOOT=yes
      ```

   iii. Configure the remaining entries (IP address, gateway, DNS, and domain) according to your network configuration.

   iv. Save the file.
6. Add physical port eth0 to virtual bridge virbr0:

   i. Open the interface configuration file for the eth0 port in the vi editor (or other similar editor):

      vi /etc/sysconfig/network-scripts/ifcfg-eth0 ↵

   ii. Add physical port eth0 to virbr0 by adding the following attributes to the file:

      ```
      BRIDGE=virbr0
      ```

   iii. Remove the IP address from the eth0 configuration file if it has been assigned to the bridge.

   iv. Save the file.

7. Adjust the MTU of the bridge by changing the MTU of the eth0 port:

   ip link set eth0 mtu 9000 ↵

8. Restart the service network:

   systemctl restart network ↵

9. Verify the link status of the eth0 interface:

   ethtool eth0 ↵

   Output example:

   [root@host ~]# ethtool eth0

   Settings for eth0:

   ...

   Link detected: yes

6 — Customizing Initial Configuration of the VSR

6.1 — In This Chapter

This chapter describes the procedures for customizing the initial configuration of a VSR virtual machine (VM).

Topics in this chapter include:

- VSR VM Configuration Using SMBIOS
- Modifying Simulated HW Configuration of the VSR-SIM

6.2 — VSR VM Configuration Using SMBIOS

Every VSR VM boots from the same set of software images stored on a QCOW2 or VMDK virtual disk. The unique characteristics of the VM—for example, its system type (VSR-SIM or VSR-RR) and its role in that system (for example, CPM vs. IOM in a VSR-SIM)—is determined at boot time based on the SMBIOS information passed from the hypervisor to the VM. This SMBIOS information is encoded as a text string in the following format:

```
TIMOS:<key1>=<value1> <key2>=<value2> …
```

where:

`<keyx>`—name of a parameter (typed without the angle brackets)

`<valuex>`—value of the parameter to the left of the equal/assignment operator

Table 6.1 lists the supported <key> parameters and their purpose.

Table 6.1 — <key> Parameters

Parameter	Description

address	Sets the management IP address of the active and/or standby CPM
	For example:
	`address=192.0.2.1/24@active`
	`address=192.0.2.2/24@standby`
	A management IP address (in the same IP subnet) should be assigned to both CPMs in a redundant VSR-SIM system
	Note—The VSR-RR does not have a standby CPM

static-route	Specifies one or more static routes associated with the management router in the following format: <IP prefix>/<prefix-length>@<next-hop-ip> For example: `static-route=192.0.0.0/16@192.0.2.254`

license-file	Specifies the location of the license file for a VSR-RR or the CPM VM of a VSR-SIM
	For example:
	`license-file=ftp://<username>:<passwd>@host/path/`
	`license-file.txt`
	`license-file=cf3:/license-file.txt`

Table 6.1 — <key> Parameters (Continued)

Parameter	Description

primary-config	Specifies the location of the primary configuration file for a VSR-RR or a CPM VM of a VSR-SIM
	For example:
	`primary-config=ftp://<username>:<passwd>@host/path/config.txt`
	`primary-config=cf3:/config-alternate.txt`

slot	Specifies the emulated slot ID associated with the VSR-SIM VM. Valid values depend on the chassis type
	Note—This parameter is applicable to the VSR-SIM only
	For example:
	`slot=A`
	`slot=2`

chassis	Specifies the emulated chassis type associated with the VSR-SIM VM
	Note—This parameter is applicable to the VSR-SIM only
	For example:
	`chassis=SR-c12`
	`chassis=SR-7`
	`chassis=SR-12`
	`chassis=SR-a4`
	`chassis=XRS-20`

card	Specifies the emulated card type associated with the VSR-SIM VM
	Note—This parameter is applicable to the VSR-SIM only
	For example:
	`card=cfm-xp`
	`card=cpm-x20`
	`card=cpm5`
	`card=iom3-xp-b`
	`card=xcm-x20`

mda/<number>	Specifies the emulated MDA type in the MDA slot <number>. Depending on the VSR-SIM chassis type, this parameter may only be supported when the slot number and card type correspond to an IOM VM
	Note—This parameter is applicable to the VSR-SIM only
	For example:
	`mda/1=m5-1gb-sfp-b`
	`mda/2=isa-tunnel`

6.2.1 — SMBIOS Text String Examples

This section provides examples of SMBIOS text strings Virtualized Service Router VSR.

Example 1

```
TIMOS:address=10.0.0.1/24@active static-route=10.0.0.0/
8@10.0.0.254 license-file=ftp://user1:secret@172.16.0.1/
home/vsr-rr-license-02367122.txt
```

This SMBIOS text string causes the VM to boot up as a VSR-RR with a management IP address of 10.0.0.1. It has a static route via the gateway 10.0.0.254 in order to reach other 10/8 addresses. The VSR-RR is instructed to look for its license file on the 172.16.0.1 FTP server.

Example 2

```
TIMOS:address=1.1.1.1/24@active 1.1.1.2/24@standby static-
route=1.1.0.0/16@1.1.1.100 license-file=cf3:/license.txt
slot=A chassis=SR-7 card=cpm5
```

This SMBIOS text string causes the VM to boot up as a CPM virtual machine of a VSR-SIM emulating an SR-7 chassis. The emulated card type is CPM5. The management IP address is either 1.1.1.1 or 1.1.1.2, depending on whether CPM A is active or standby. A static route has been installed to reach 1.1/16 addresses via next-hop router 1.1.1.100. The VSR-SIM is instructed to look for its license file on its local CF3 disk.

Note: The maximum length of an SMBIOS text string is 1024 characters.

6.2.2 — Providing SMBIOS Information to the VM

The following options are supported for providing the SMBIOS text string to the VM so that it is available at bootup:

- Passing Information in the libvirt Domain XML File for the VM
- Setting the machine.id Property of a VMware VM

6.2.2.1 — Passing Information in the libvirt Domain XML File for the VM

The SMBIOS text string appears as the 'product' entry for the system, as illustrated in the sample **libvirt** domain XML file below. The SMBIOS string in this XML example corresponds to Example 1 described in section 6.2.1, SMBIOS Text String Examples.

```
<domain type='kvm'>
  --- SNIP ---
  <os>
    <type arch='x86_64' machine='rhel6.0.0'>hvm</type>
    <smbios mode='sysinfo'/>
  </os>
  <sysinfo type='smbios'>
    <system>
           <entry name='product'> TIMOS:address=10.0.0.1/24@active static-
route=10.0.0.0/8@10.0.0.254 license-file=ftp://user1:secret@172.16.0.1/home/vsr-
rr-license-02367122.txt
           </entry>
    </system>
 </sysinfo>
```

6.2.2.2 — Setting the machine.id Property of a VMware VM

The machine.id property can be set using the vSphere Web client or by manually editing the VMX configuration file associated with the VM.

A sample VMware VMX file is shown below. The SMBIOS string in this XML example corresponds to Example 1 described in section 6.2.1, SMBIOS Text String Examples.

```
.encoding = "UTF-8"
config.version = "8"
virtualHW.version = "10"
virtualHW.productCompatibility = "hosted"
vmci0.present = "TRUE"
vmci.filter.enable = "true"
displayName = "VSR-RR1"
extendedConfigFile = "VSR-RR1.vmxf"
--- SNIP ---
uuid.action = "keep"
uuid.bios = "42 27 69 10 b4 9f 2c a3-d2 e2 e9 49 22 e4 18 0f"
vc.uuid = "50 27 46 1b 65 d9 0d e6-50 b3 62 4f 09 0a bd 85"
machine.id = "TIMOS:address=10.0.0.1/24@active static-route=10.0.0.0/8@10.0.0.254
license-file=ftp://user1:secret@172.16.0.1/home/vsr-rr-license-02367122.txt"
```

6.3 — Modifying Simulated HW Configuration of the VSR-SIM

This section describes the **chassis**, **slot**, **card**, and **mda** parameters of the SMBIOS text string. These parameters are only applicable to the VSR-SIM and are used to modify the VSR-SIM's simulated hardware configuration. See section 6.2 for detailed information about SMBIOS.

Caution: You must ensure that the content and syntax of the SMBIOS text string are correct. If there are errors in the SMBIOS text string, the system may revert to the default VSR-SIM configuration and reboot automatically after 60 minutes if steps are not taken to specify a valid **license-file**. See Section 9.5, Verifying VSR Installation for more information.

Table 6.2 summarizes the valid values for each of the configurable attributes that affect the VSR-SIM hardware configuration.

Table 6.2 — VSR-SIM Hardware Configuration Options

Attribute	Description	Valid Values
slot=	The slot identifier of the card represented by this VM	Chassis-dependent • A, B, C, and D identify CPM slots C and D are CPM slots for the 7950 XRS • 1 to 20 identify IOM/XCM slots
chassis=	The type of chassis virtualized by the VSR-SIM	One of the following: • SR-c4 • SR-c12 • SR-a4 • SR-a8 • SR-7 • SR-12 • SR-12e • XRS-16 • XRS-20

Table 6.2 — VSR-SIM Hardware Configuration Options (Continued)

Attribute	Description	Valid Values
card=	The type of card represented by this VM; uses card-type values from SR OS CLI	Chassis-dependent Examples: • cfm-c4-xp • cfm-xp • cfm-xp-b • sfm3-7 • sfm3-12 • sfm4-7 • sfm4-12 • sfm4-12e • cpm5 • cpm-x20 • xcm-x20 • iom3-xp • iom3-xp-b • imm1-100gb-cfp • imm12-10gb-sf+
mda/*n*=	If a card supports one or more MDAs (XMAs), this parameter indicates the type of MDA installed in each slot (n). Uses mda-type values from SR OS CLI.	Card-dependent Examples: • m5-1gb-sfp-b • m1-10gb-xp-xfp • m12-1gb+2-10gb-xp • m20-1gb-sfp • isa-aa • isa-bb • isa-tunnel • cx20-10g-sfp • cx20-100g-sfp • x40-10g-sfp • x4-100g-cxp • cx6-40g-qsfp • cx72-1g-csfp • x4-100g-cfp2

6.3.1 — VSR-SIM SMBIOS Known Limitations

The following are known limitations with the VSR-SIM hardware configuration options.

- on the 750 SR-c4 chassis, the integrated MDA should be defined in slot 5; mda/5=icm2-10gb-xp-xfp. It is automatically provisioned.
- one ISA MDA is supported per IOM
- isa-bb is not supported as a hardware configuration for FP3+ based IOMs (for example, imm-2pac-fp3, iom4-e)
- imm-1pac-fp3 IOM can only be used in combination with p160-1gb-csfp MDA
- isa-tms is not supported.
- isa-video is not supported
- export restriction ISA flavors are not supported

6.3.2 — VSR-SIM SMBIOS Examples

In this example, the VSR-SIM deployment simulates an SR-7 chassis with three cards, as follows:

- SFM3-7 in slot A
- SFM3-7 in slot B
- IOM3-XP in slot 1

  The IOM3-XP card has two MDA slots: one is populated with an ms-isa (isa-tunnel personality) and the other is equipped with a 1-port 10GE MDA.

The total system requirement for this example is three VMs. The hardware configuration portion of each VM's SMBIOS text string is as follows:

Virtual machine 1 (CPM A)

```
TIMOS:slot=A chassis=SR-7 card=sfm3-7 [other parts of the SMBIOS string omitted]
```

Virtual machine 2 (CPM B)

```
TIMOS:slot=B chassis=SR-7 card=sfm3-7 [other parts of the SMBIOS string omitted]
```

Virtual machine 3 (IOM 1)

```
TIMOS:slot=1 chassis=SR-7 card=iom3-xp mda/1=isa-tunnel mda/2=m1-10gb-xfp [other
parts of the SMBIOS string omitted]
```

7 — Deploying the VSR VM on VMware ESXi Hosts

7.1 — In This Chapter

This chapter describes how to deploy the VSR virtual machines (VMs) on VMware ESXi hosts using the vSphere Web Client interacting with the vSphere vCenter server.

Note: This chapter provides instructions on managing VMs on VMware ESXi hosts using the vSphere Web Client only. Techniques for managing VMs on ESXi hosts using other options (for example, vSphere Windows client, direct ESXi shell access) are beyond the scope of this guide.

Before you can perform the procedures in this chapter, you must install the recommended server hardware and server OS, as described in Chapter 3, Hypervisor Requirements.

The topics in this chapter include:

- Creating VSR VMs using the vSphere Web Client GUI

Note: Unless otherwise noted, the graphics in this chapter show sample installations and configurations.

7.2 — Creating VSR VMs using the vSphere Web Client GUI

Perform this procedure to create a new VSR VM using the vSphere Web Client.

Note: This procedure assumes that you have already installed the vCenter Server and added the ESXi host to the data center group.

1. Connect to the vCenter server over HTTP and log in from the VMware vCenter Single Sign-On screen, as shown in Figure 7.1.

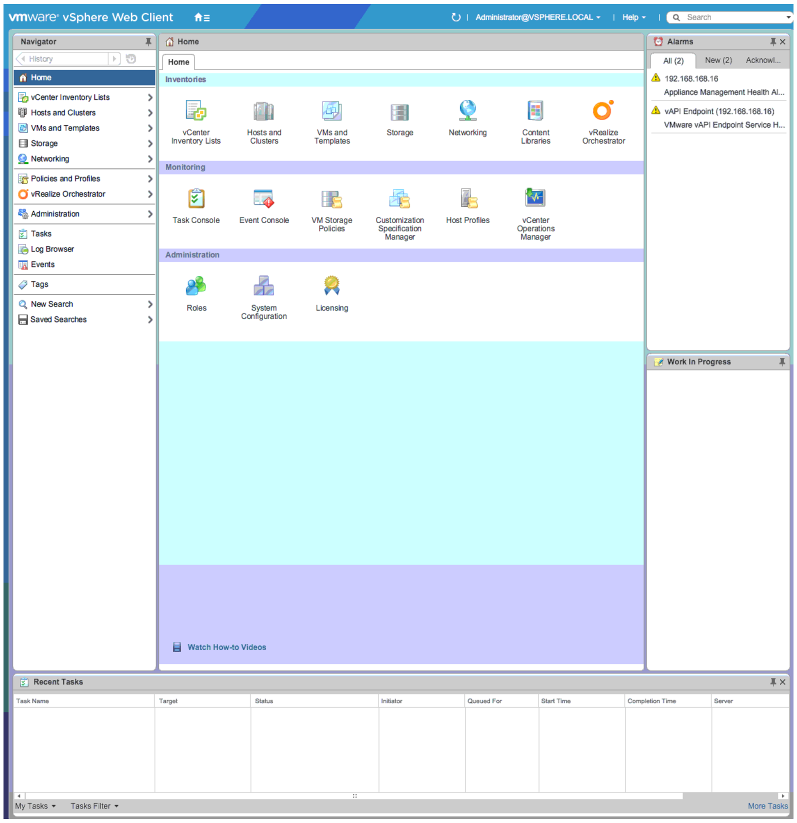
i. Enter the username.

ii. Enter the password you set during installation.

Figure 7.1 — VMWARE vCenter Server Login Screen

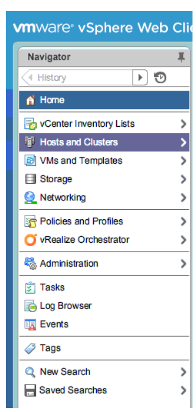
2. Click Login. The vSphere Web Client dashboard is displayed, as shown in Figure 7.2.

Figure 7.2 — vSphere Web Client Dashboard



3. From the Navigator pane, choose Home→Hosts and Clusters, as shown in Figure 7.3.

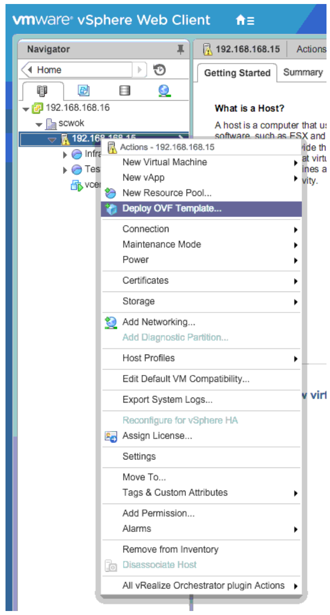
Figure 7.3 — Home Menu



4. Select and deploy your OVF template as follows:
   i. Browse to the source location of your data center.
   ii. Right-click the ESXi host on which to deploy the VSR.
   The Actions menu is displayed, as shown in Figure 7.4.

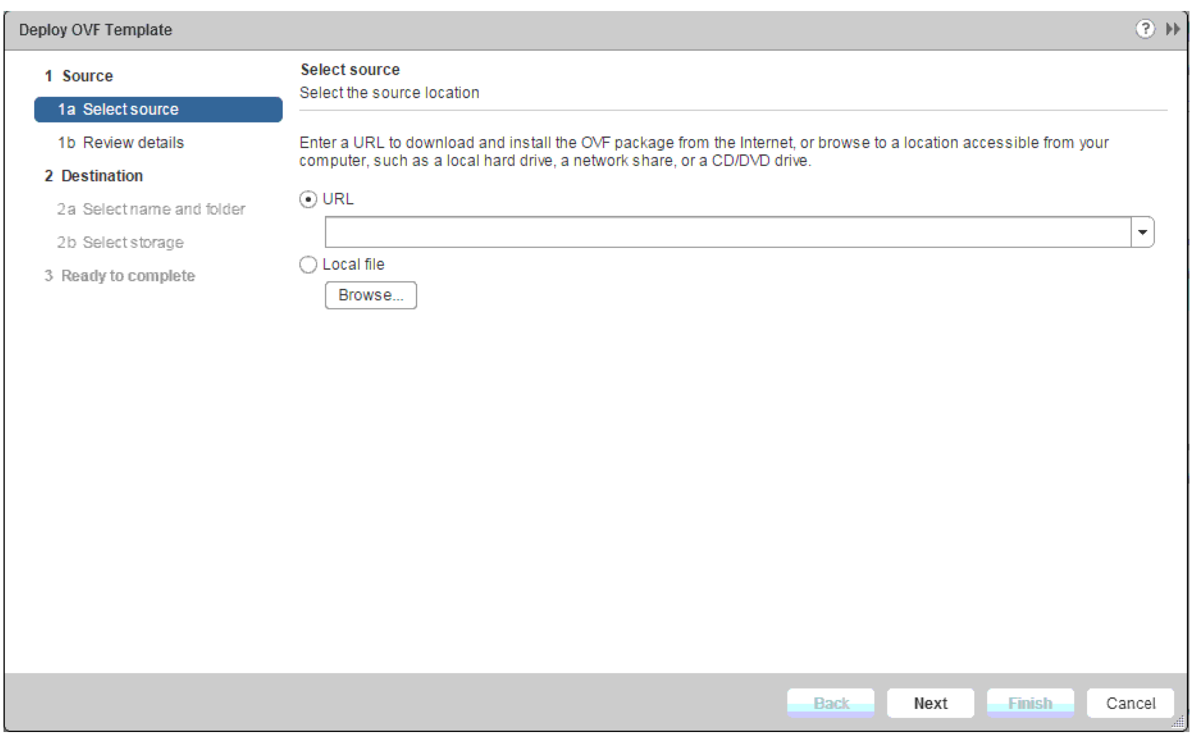
iii. Select the Deploy OVF Template option.

Figure 7.4 — Actions Menu



5. The Deploy OVF Template window appears with the Select Source option selected, as shown in Figure 7.5

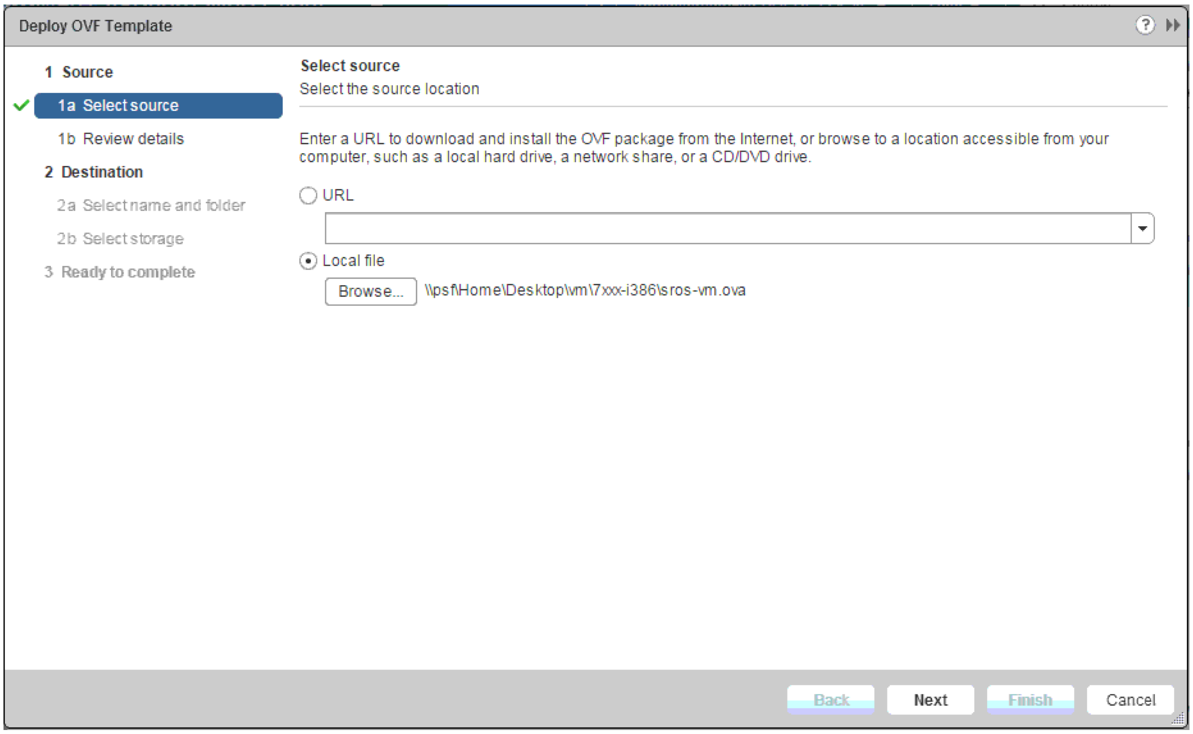
Figure 7.5 — Deploy OVF Template Window



6. Specify the location of the VSR OVA archive file. Perform one of the following steps:
   a. Select the URL option to download the OVF package from the Internet.
   b. Select the Local File option to browse location and retrieve a local file.

   Figure 7.6 shows an example selection using the Local File option.

Figure 7.6 — Local File Option

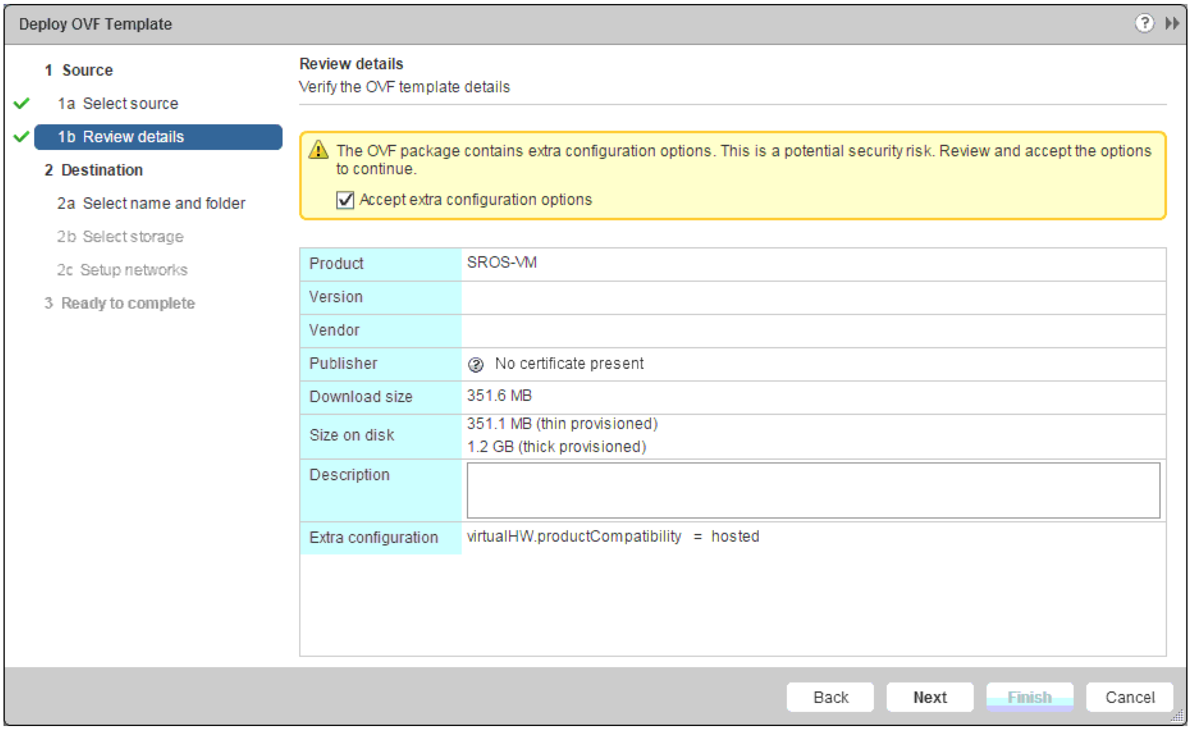


Click Next to advance to the next screen.

7. From the Review Details pane, select the **Accept Extra Configuration Options** check box, as shown in Figure 7.7.

   This option allows you to accept the additional configuration options in the VSR OVA archive file that are not available in the standard VMware OVA templates.

Figure 7.7 — Review Details



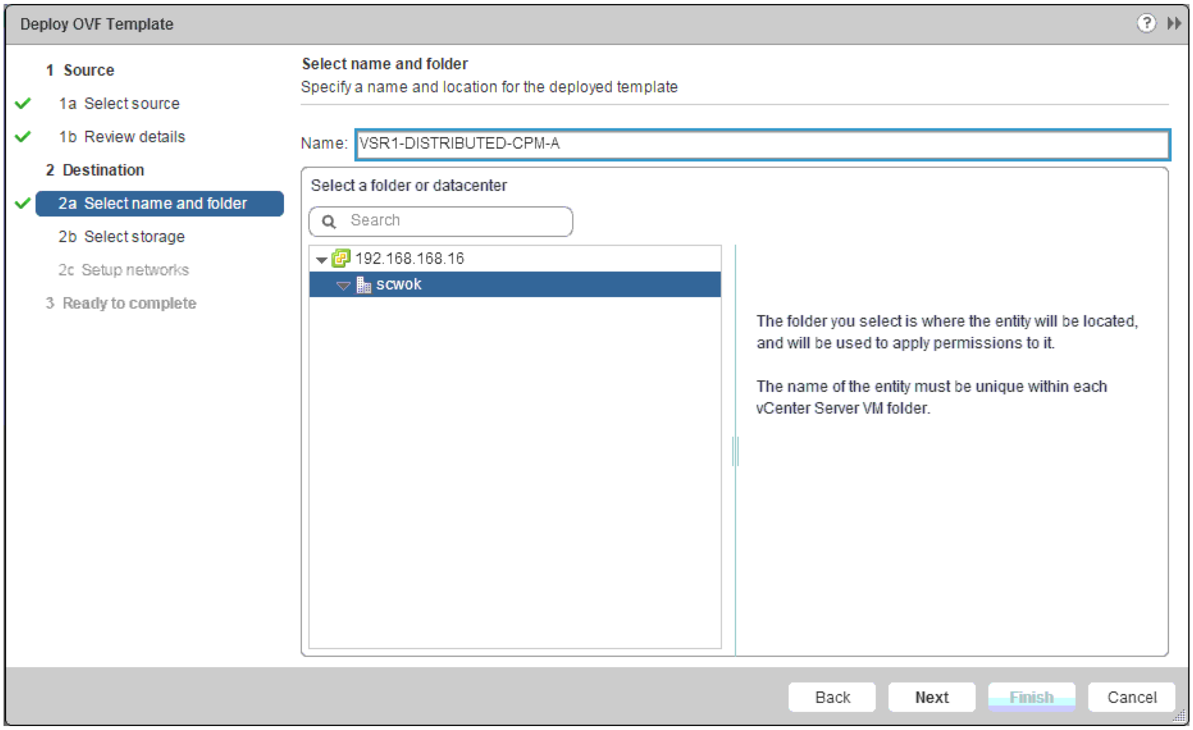
Click next to advance to the next screen.

8. Specify a name and location for the deployed OVF template, as shown in Figure 7.8.

   i. Name the VSR virtual machine.

      The name can be up to 80 characters and must be unique within the vCenter Server VM folder. The example VM in Figure 7.8 is the first CPM in a distributed model VSR and is named VSR1-DISTRIBUTED-CPM-A.

Figure 7.8 — Name the OVF Template



ii. Select the data center where the VSR will be deployed.

iii. Click Next to advance to the next screen.

9. Select the storage location for the deployed template, as shown in Figure 7.9.

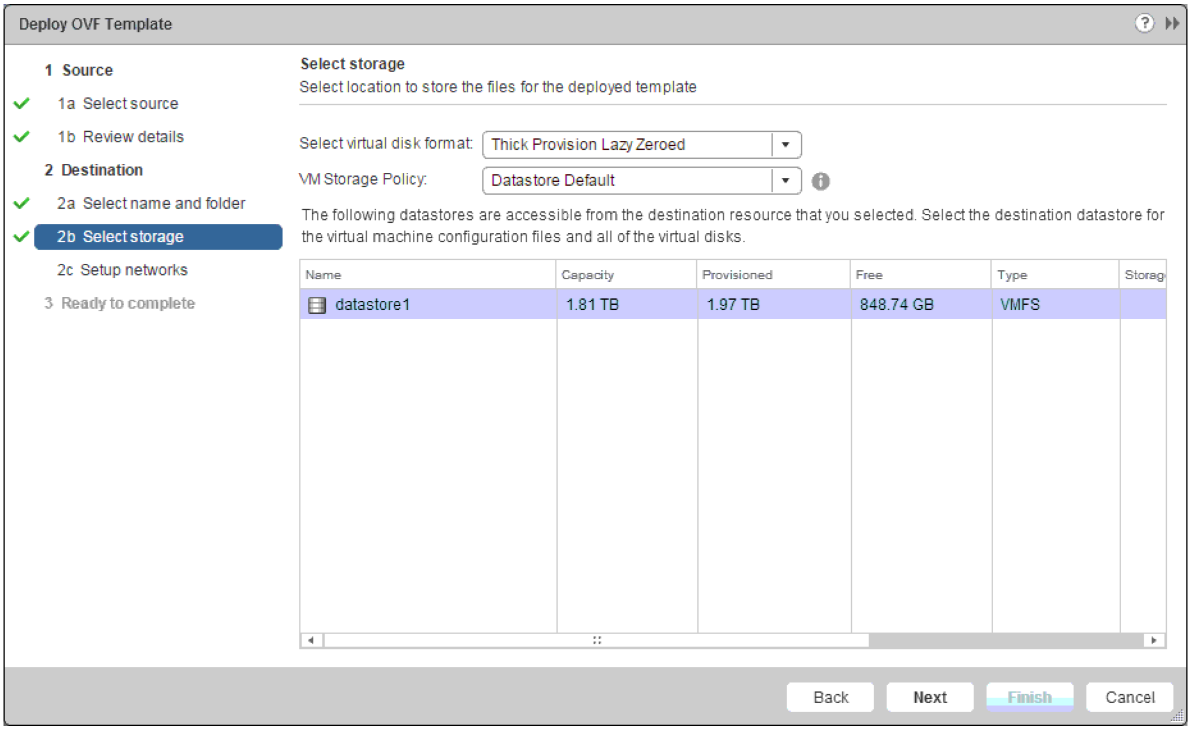
i. Select a virtual disk format.

ii. Select a VM storage policy.

iii. Click Next to advance to the next screen

For more information about the right virtual disk format for your configuration, go to www.vmware.com.

Figure 7.9 — Storage Location Selection



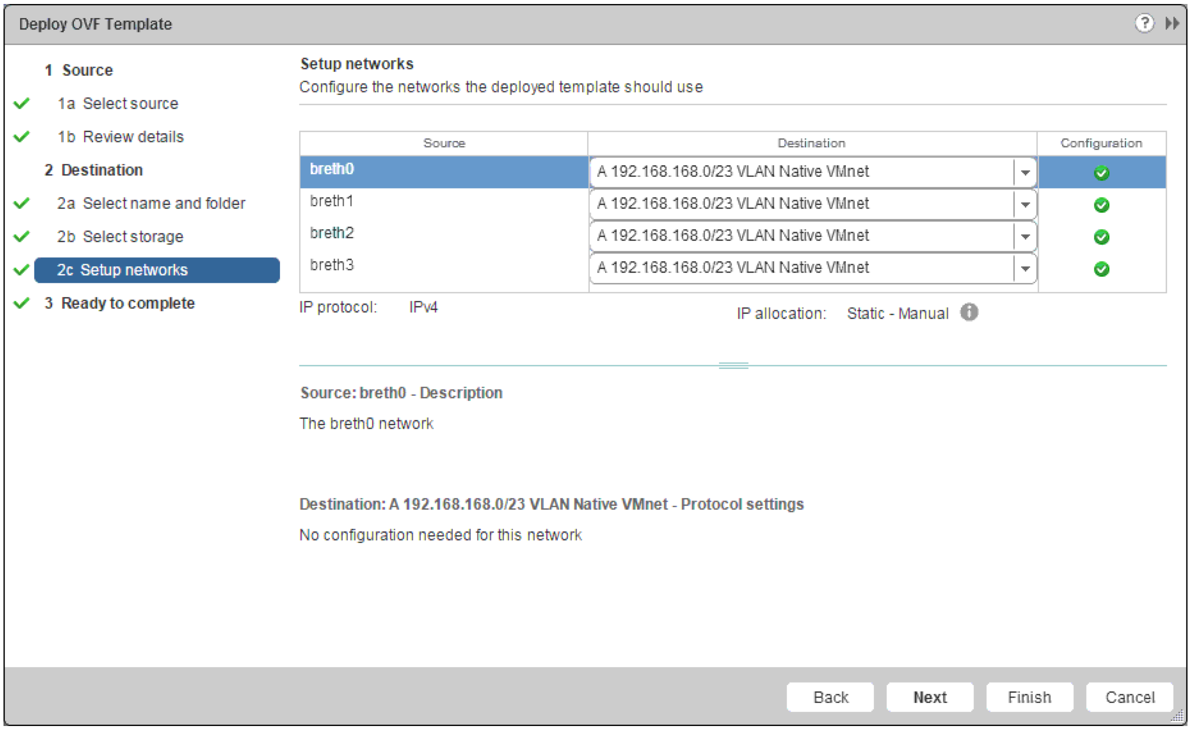
10. Configure the networks for your OVF template, as shown in the example in Figure 7.10.

Note: Alcatel-Lucent recommends that you map network interfaces carefully. Ensure that Network Interface ‘breth0’ is mapped to the correct destination network as this will become a management interface of the system.

By default, the VSR is deployed with four network interfaces: breth0, breth1, breth2, and breth3. The first interface (breth0) corresponds to a management interface. The second interface (breth1) corresponds to the inter-VM control/data plane. The third and fourth interfaces (breth2 & breth3) do not have a function in a distributed model CPM VM and can be ignored in this example. See Chapter 5, VSR Network Connectivity for more information.

Click Next to advance to the next screen.

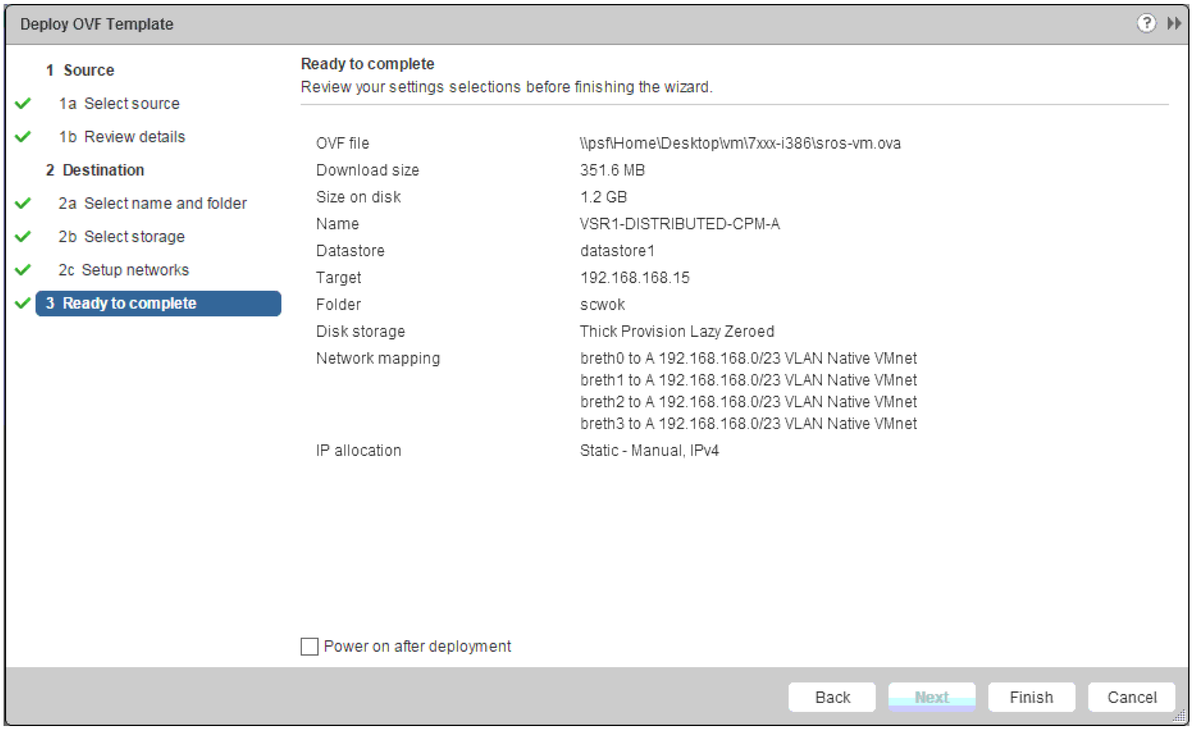
Figure 7.10 — Mapping Network Interfaces



11. Review your settings selections.

Note: Ensure that the ‘Power on after deployment’ option is not selected, as shown in Figure 7.11.

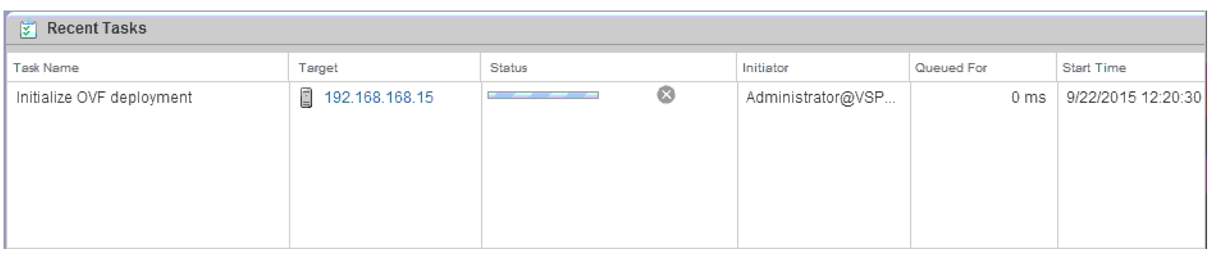
Figure 7.11 — Review OVF Settings



12. Click Finish to deploy the VSR.

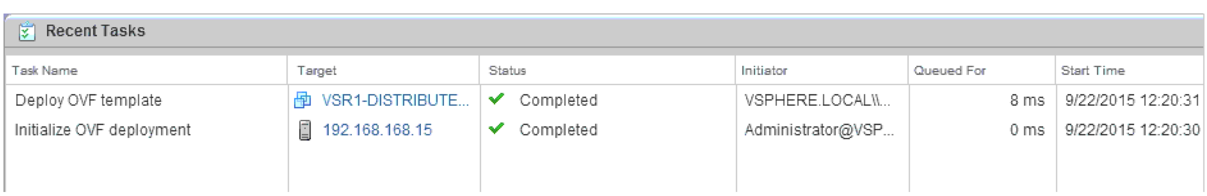
The Recent Tasks section of the dashboard displays the status of the VSR deployment, as shown in Figure 7.12.

Figure 7.12 — Deploying the OVF



An sample completed deployment is shown in Figure 7.13.

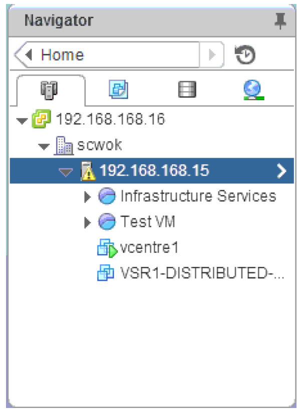
Figure 7.13 — Completed OVF Deployment



13. Configure the memory and resource allocation for the VSR Virtual Machine:

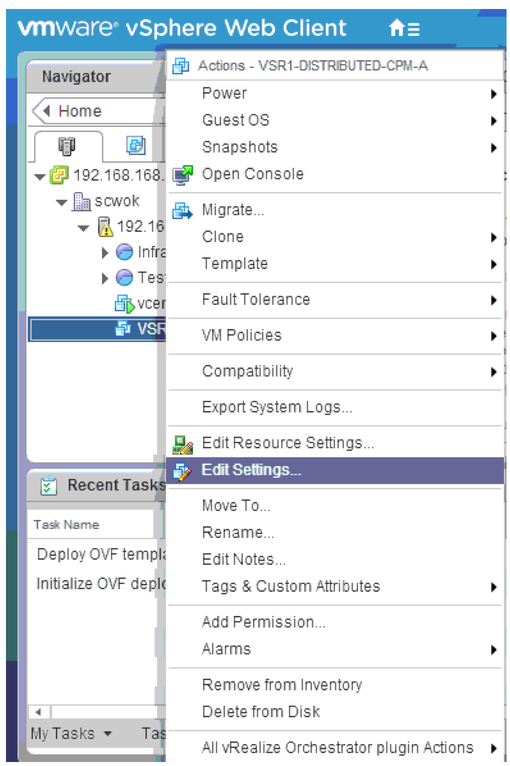
i. In the Navigator Pane. locate the newly deployed VSR, as shown in Figure 7.14

Figure 7.14 — Navigator Pane



ii. Right-click the VSR name and select **Edit Settings**, as shown in Figure 7.15

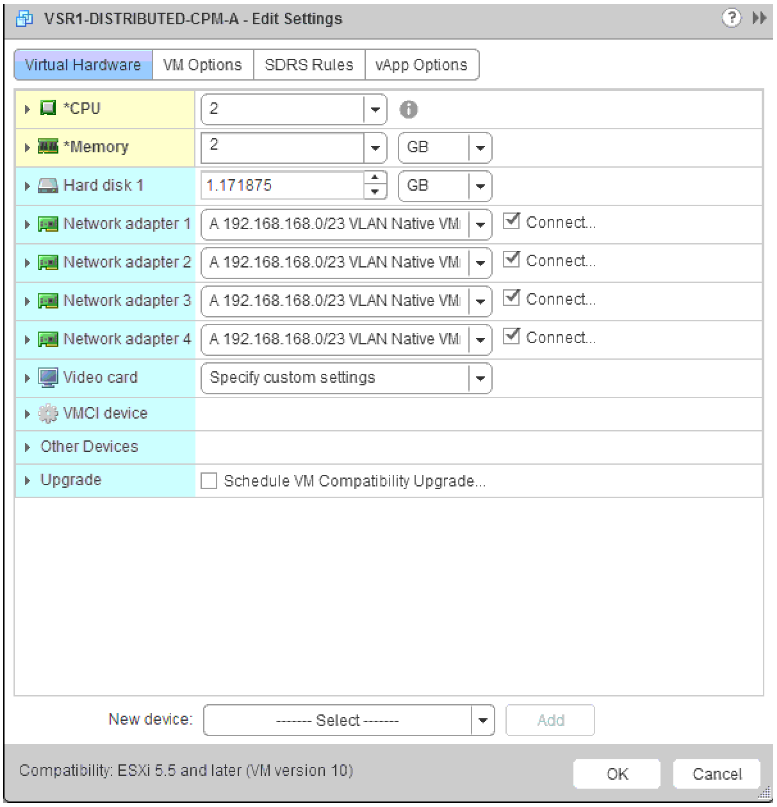
Figure 7.15 — Actions Menu



14. The Edit Settings window appears with the Virtual Hardware tab selected, as shown in Figure 7.16.

Select the number of vCPU and vMemory to allocate to the VSR VM.

Figure 7.16 — Virtual Hardware Options



15. Configure a serial port to obtain console connectivity to the VSR:

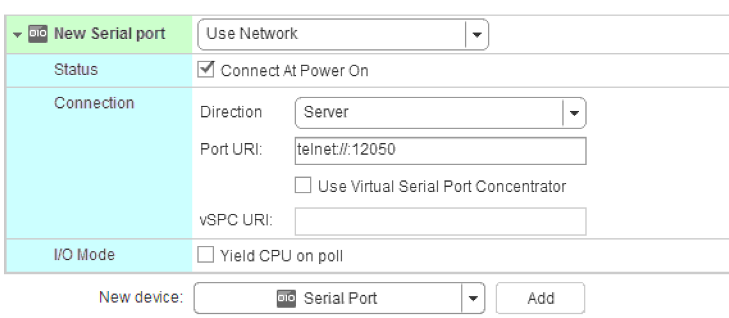
i. Click the **New Device** drop-down list option located towards the bottom of the Edit Settings window, shown in Figure 7.16.

ii. Select a Serial Port from the drop-down list.

iii. Configure the serial port, as required.

The sample configuration shown in Figure 7.17 enables console access to the VSR via telnet by routing TCP port 12050 on the ESXi host to the serial port on the VSR. Other methods are available but are not documented here.

Figure 7.17 — Serial Port Configuration



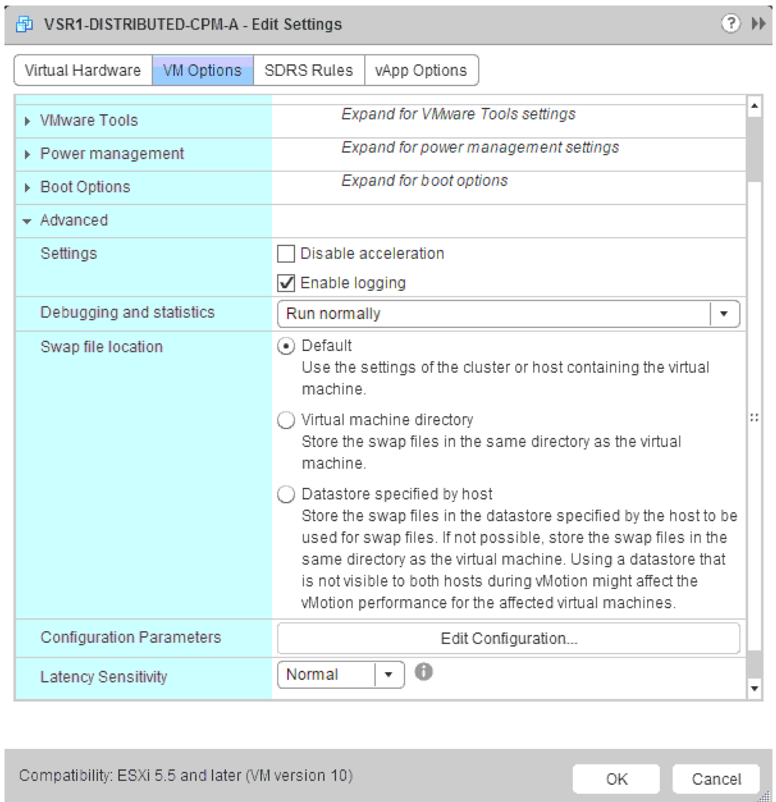
iv. Click Add to implement the new serial port configuration.

16. Add and modify the VSR configuration parameters:

i. From the Edit Settings window, select the VM Options tab.

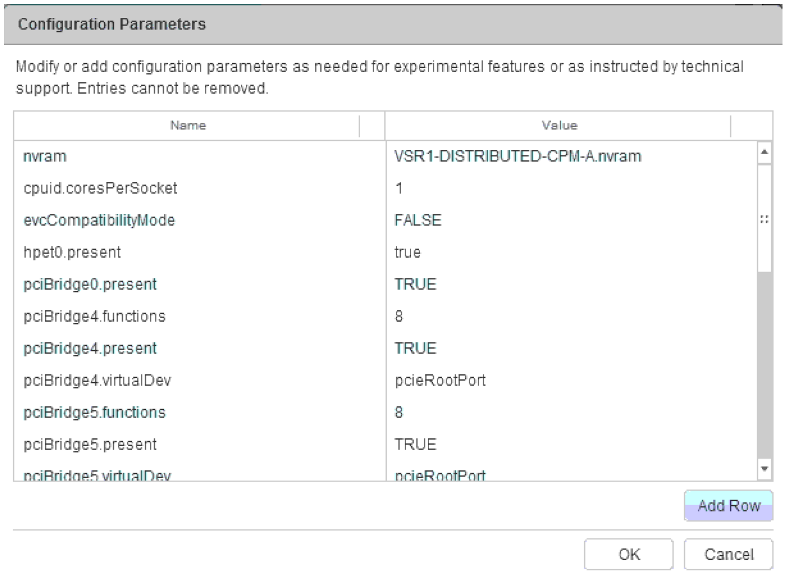
ii. Click the arrow to expand the Advanced Settings section, as shown in Figure 7.18

Figure 7.18 — Advanced Settings



iii. Click the Edit Configuration button to display the Configuration Parameters window, as shown in Figure 7.19

Figure 7.19 — Configuration Parameters Window



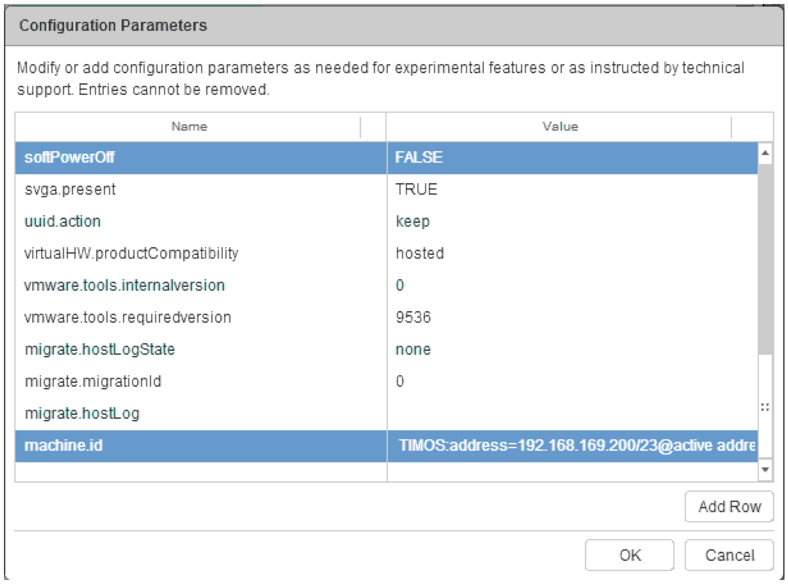
iv. Click the Add Row button to add a machine.id setting.

Add a machine.id setting and set its value to the SMBIOS text string appropriate for the VM. See Chapter 6, for further details.

Figure 7.20 shows an example VSR-SIM in a distributed mode acting as CPM-A in an SR-12. The example machine.id setting is as follows:

```
TIMOS:address=192.168.169.200/23@active
address=192.168.169.201@standby license-file=ftp://
user:password@192.168.169.110/license.txt slot=A
chassis=SR-12 card=sfm4-1
```

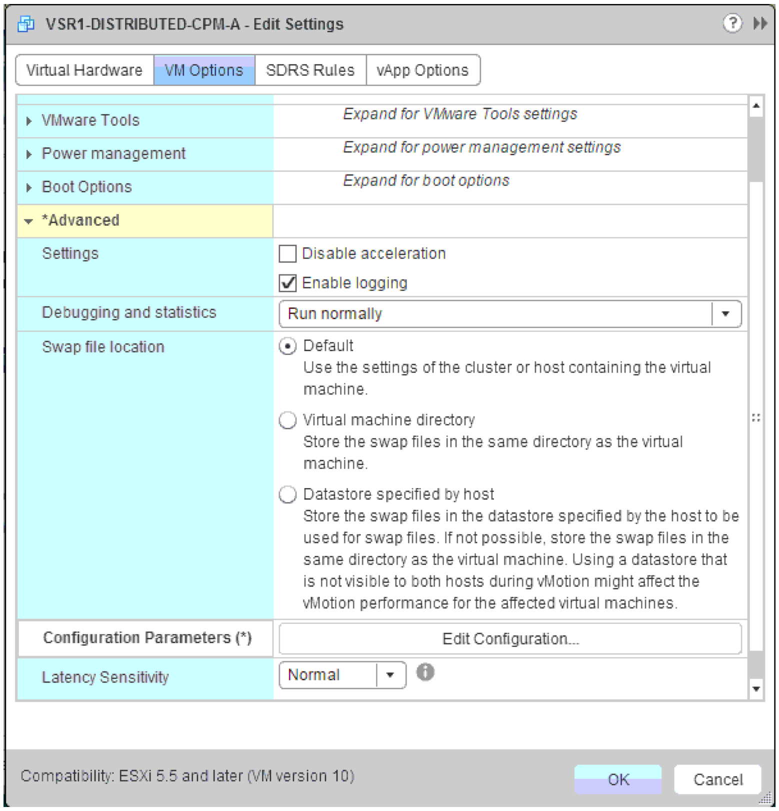
Figure 7.20 — Example machine.id Setting



v. Click OK to return to the Edit Settings window.

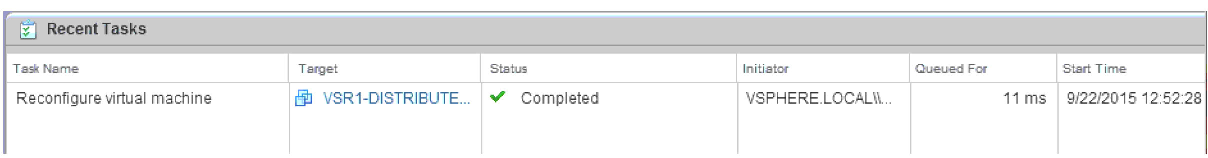
17. From the Edit Settings window, click OK to finish the VSR VM configuration, as shown in Figure 7.21.

Figure 7.21 — Edit Setting Window



18. From the Recent Tasks window, check the status of the VSR VM reconfiguration. A sample completed configuration is shown in Figure 7.22.

Figure 7.22 — Completed Task



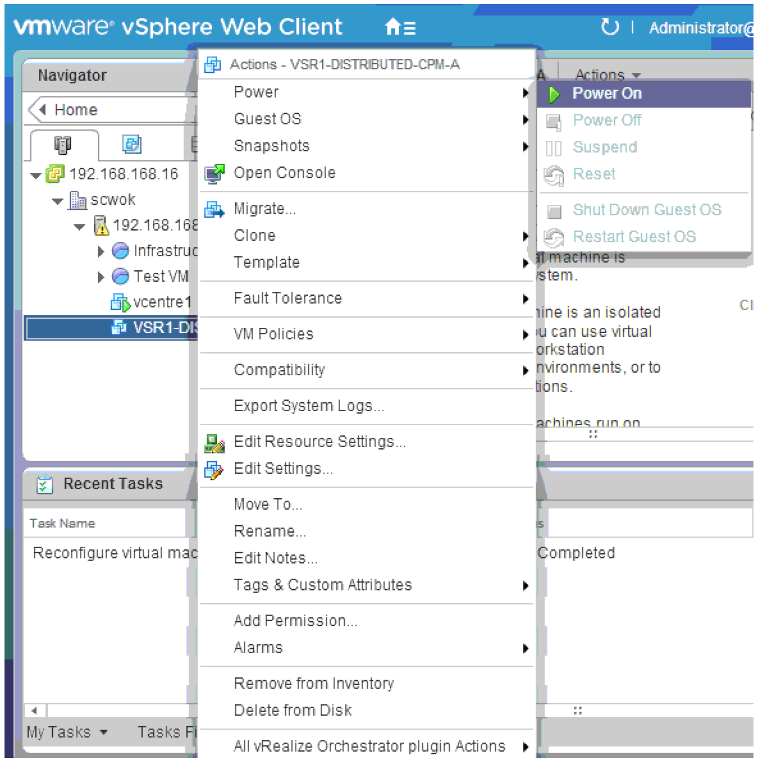
19. Start the VSR:

i. Locate the VSR1-DISTRIBUTED-CPM-A VSR in the Navigator window and right click it.

The Actions menu appears.

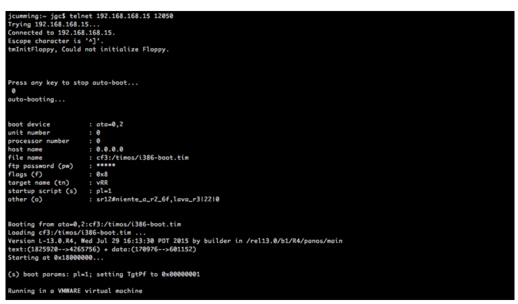
ii. Select Actions→Power→Power On, as shown in Figure 7.23. The VSR starts the boot process.

Figure 7.23 — Power On the VSR



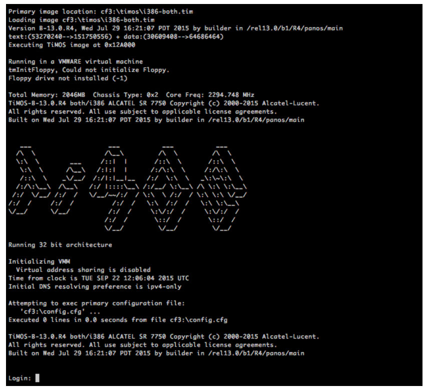
20. You can connect to the console by telnetting to the ESXi host (in this example 192.168.168.15) on the TCP port number you defined earlier (in this example 12050), as shown in Figure 7.24.

Figure 7.24 — Telnet Screen



21. The login screen is presented once the VSR is up and running successfully, as shown in Figure 7.25.

Figure 7.25 — Login Screen



22. Login using the following credentials:

login: admin

password: admin

SROS commands can now be issued as normal.

Note: As this is an example of a distributed VSR-SIM, you will need to create another VM for an IOM to forward traffic and provision it using the SROS CLI.

8 — Deploying the VSR VM on Linux Hosts Using the KVM/QEMU Hypervisor

8.1 — In This Chapter

This chapter describes how to deploy the VSR virtual machines (VMs) on host machines using the KVM/QEMU hypervisor.

Note: This guide provides instructions on managing the KVM/QEMU VMs using the open-source **libvirt** package. OpenStack and other IAAS (Infrastructure as a Service) cloud management systems can also be used as management tools; however, deployment instructions using these systems are beyond the scope of this guide.

The topics in this chapter include:

- The libvirt Domain XML Configuration File
- VM Administration Using Virtual Shell Commands

The **libvirt** package is used to define VM properties and to perform administrative actions on KVM/QEMU VMs to support the VSR functionality. It provides the **virsh** command line application that is referenced in this guide.

Note: It is recommended (although not mandatory) that you deploy all the virtual machines of one distributed model VSR-SIM on hosts running the same type of hypervisor, such as KVM/QEMU.

You must complete the following tasks before performing the procedures described in this chapter:

- install the recommended server hardware, Linux OS, and hypervisor, as described in Chapter 3, Hypervisor Requirements
- configure the system interfaces, as described in Chapter 5, VSR Network Connectivity

8.2 — The libvirt Domain XML Configuration File

When you use **libvirt** to manage KVM/QEMU virtual machines used for VSR deployment, the properties of each VM (such as the amount of memory assigned to the guest and the number of virtual CPU cores allocated to the VM), are encoded in a domain XML file.

8.2.1 — Sample libvirt XML File

This section provides a sample **libvirt** XML file for a VSR-RR VM.

The VSR-RR virtual machine configuration is as follows:

- 4 GB of memory allocation
- 8 virtual CPUs
- a single disk backed by a QCOW2 disk image stored locally; in the VSR context, this disk (mapped as HDA) operates as the CF3: device

The VSR-RR has two vNIC interfaces as follows:

- the first interface (corresponding to port A/1) is attached to a Linux bridge called br0
- the second interface (corresponding to port 1/1/1) is directly connected (in pass-through mode) to the host NIC port eth3

VirtIO is used for both vNIC interfaces.

Note: When you create your VSR-RR VM, substitute the parts of the sample **libvirt** file represented in red font with values particular to your setup and environment.

```
<domain type='kvm'>
  <name>rr1</name>
  <uuid>4dea22b3-1d52-d8f3-2516-782e-98ab3fa0</uuid>
  <memory>4194304</memory>
  <currentMemory>4194304</currentMemory>
  <cpu mode='custom' match='minimum'>
    <model>SandyBridge</model>
    <vendor>Intel</vendor>
  </cpu>
  <vcpu current='8'>8</vcpu>
  <os>
    <type arch='x86_64' machine='rhel6.0.0'>hvm</type>
    <smbios mode='sysinfo'/>
  </os>
  <sysinfo type='smbios'>
    <system>
<entry name='product'>TIMOS:address=192.0.2.1/32@active static-route=192.168.12.0/
24@192.0.2.254 license-file=ftp://user:pass@192.168.12.30/~/dir/filename.txt</
entry>
```

```
    </system>
  </sysinfo>
  <clock offset='utc'/>
    <timer name='pit' tickpolicy='delay'/>
    <timer name='rtc' tickpolicy='delay'/>
  </clock>
  <devices>
    <emulator>/usr/libexec/qemu-kvm</emulator>
    <disk type='file' device='disk'>
      <driver name='qemu' type='qcow2' cache='none'/>
      <source file='/var/lib/libvirt/images/sros-vm.qcow2'/>
      <target dev='hda' bus='virtio'/>
    </disk>
    <interface type='bridge'>
      <mac address='FA:AC:A6:02:01:00'/>
      <source bridge='br0'/>
      <model type='virtio'/>
    </interface>
    <interface type='direct'>
      <mac address='FA:AC:A6:02:01:01'/>
      <source dev='eth3' mode='passthrough'/>
      <model type='virtio'/>
    </interface>
    <console type='tcp'>
      <source mode='bind' host='0.0.0.0' service='2500'/>
      <protocol type='telnet'/>
      <target type='virtio' port='0'/>
    </console>
  </devices>
  <seclabel type='none'/>
</domain>
```

8.2.2 — Sample libvirt: Key Elements Explained

This section provides a brief explanation of the key portions of the sample **libvirt** domain XML file described in section 8.2.1.

General XML Elements

The following XML statement indicates that the VM should be instantiated using KVM as the hypervisor.

```
<domain type='kvm'>
```

The following XML statement specifies the name of the VM; a maximum of 128 characters are permitted. The **virsh** commands make reference to this name.

```
<name>rr1</name>
```

Memory Allocation

The following XML statements specify that the VM should be allocated 4 914 304 KB of memory. This is equivalent to 4 GB, which is the minimum amount of memory that you can allocate to a VSR VM.

```
<memory>4194304</memory>
<currentMemory>4194304</currentMemory>
```

The following block of XML statements specifies that the CPU presented to the guest should provide a minimum of Intel Sandy Bridge (or later) capabilities. The VM should not be instantiated if the host CPU is not Sandy Bridge or a more recent CPU. The last line indicates that the VM should be allocated 8 virtual CPU cores. There are no constraints on the mapping of virtual CPU cores to actual physical CPU cores and threads, that is, no CPU pinning. In general, CPU pinning is not required for the VSR-RR and VSR-SIM.

```
<cpu mode='custom' match='minimum'>
  <model>SandyBridge</model>
  <vendor>Intel</vendor>
</cpu>
<vcpu current='8'>8</vcpu>
```

OS and SMBIOS

The following block of XML statements provides details about the guest operating system. The first line indicates that the guest operating system (SR OS) is a 64-bit operating system that expects a fully virtualized x86 system.

The remaining lines provide details about the SMBIOS information that is passed to the guest when it boots up. The product entry in the SMBIOS data contains valuable information used for the initial setup and personalization of theVSR VMs. In this case, the SMBIOS text string specifies the management IP address of the VSR-RR, a static route used to reach management destinations, and the location of the license file for this VSR VM. See Chapter 6 for detailed information about SMBIOS.

```
<os>
  <type arch='x86_64' machine='rhel6.0.0'>hvm</type>
  <smbios mode='sysinfo'/>
</os>
<sysinfo type='smbios'>
  <system>
<entry name='product'>TIMOS:address=192.0.2.1/32@active static-route=192.168.12.0/
24@192.0.2.254 license-file=ftp://user:pass@192.168.12.30/~/dir/filename.txt</
entry>
  </system>
</sysinfo>
```

Devices

The following XML statement specifies that QEMU should be used for guest emulation of devices such as disks, PCI bus controllers, and NICs.

```
<emulator>/usr/libexec/qemu-kvm</emulator>
```

The following XML statements provide the guest with a QEMU-emulated IDE hard drive that stores its data in a QCOW2 formatted disk image file on the host machine. I/O communication with the disk device uses VirtIO for faster performance. The disk device that is associated with HDA maps to the CF3 device of the VSR. A second disk device called HDB (not shown in this example) may be defined; it maps to the CF1 device of the VSR.

```
<disk type='file' device='disk'>
  <driver name='qemu' type='qcow2' cache='none'/>
  <source file='/var/lib/libvirt/images/sros-vm.qcow2'/>
  <target dev='hda' bus='virtio'/>
</disk>
```

Network Interfaces

The following block of XML statements provides the guest with two vNIC interfaces.

The first vNIC interface uses VirtIO and connects to a Linux bridge called br0; this vNIC interface corresponds to a management interface of the VSR. The second vNIC interface also uses VirtIO but it is not connected to a host bridge; instead, it is directly connected to the host NIC port eth3 in pass-through mode. The second vNIC interface may be associated with an MDA I/O port of the VSR or a fabric interface, depending on the type of VSR VM.

Additional vNIC interfaces may be added to VSR VMs depending on their type and the network connectivity requirements of the system. See Chapter 5 for detailed information about vNIC interfaces.

```
<interface type='bridge'>
  <mac address='FA:AC:A6:02:01:00'/>
  <source bridge='br0'/>
  <model type='virtio'/>
</interface>
<interface type='direct'>
  <mac address='FA:AC:A6:02:01:01'/>
  <source dev='eth3' mode='passthrough'/>
  <model type='virtio'/>
</interface>
```

Console

The following XML statements provide console access to the VM. The console is available by opening a Telnet session to the host at port 2500. For example, if the host IP address is 192.0.2.1, the console of the VM can be accessed by typing telnet **192.0.2.1 2500** in a terminal window.

```
<console type='tcp'>
  <source mode='bind' host='0.0.0.0' service='2500'/>
  <protocol type='telnet'/>
  <target type='virtio' port='0'/>
</console>
```

8.3 — VM Administration Using Virtual Shell Commands

The Linux **libvirt** package provides a command-line application called Virtual Shell (**virsh**) to facilitate the administration of VMs. The **virsh** application provides commands to create and start a VM using the information contained in a domain XML file. It also provides commands to shut down a VM, list all the VMs running on a host, and output specific information about the host or a virtual machine.

Table 8.1 lists the basic virsh commands, where ***VM_name*** is the value that you configured for the **name** element in the XML configuration file. Refer to http://libvirt.org/virshcmdref.html for more information.

Table 8.1 — Basic virsh Commands

Command	Example	Result
capabilities | grep cpu	**virsh capabilities | grep cpu** ↵	Displays the number of cores on the physical machine
console	**virsh console *VM_name*** ↵	Connects the console to the VM if using the serial PTY port
define	**virsh define *VM_name*.xml** ↵	Reads the XML configuration file and creates a domain
destroy	**virsh destroy *VM_name*** ↵	Stops a VM. The VM is still available on the host and can be started again. The system status is “shut off”.
dumpxml	**virsh dumpxml *VM_name*** ↵	Displays the XML configuration file for the specified VM
list	**virsh list [--all | --inactive]** ↵	The “--all” argument displays all active and inactive VMs that have been configured and their state The “--inactive” argument displays all VMs that are defined but inactive
nodeinfo	**virsh nodeinfo** ↵	Displays the memory and CPU information, including the number of CPU cores
start	**virsh start *VM_name*** ↵	Starts the VM domain
undefine	**virsh undefine *VM_name*** ↵	Deletes a specified VM from the system

9 — Verifying and Troubleshooting the VSR Installation

9.1 — In This Chapter

This chapter describes the basic procedures for verifying your VSR virtual machine (VM) installation. Common problems that you may encounter are highlighted and possible solutions to resolve these issues are provided.

Note: This chapter provides instructions on verifying and troubleshooting VMs deployed on Linux hosts using the KVM/QEMU hypervisor only. Similar techniques can also be applied to VMs deployed in a VMware environment; however, troubleshooting the VMware environment is beyond the scope of this guide.

The topics in this chapter include:

- Verifying Host Details
- Verifying the Creation of VMs
- Verifying Host Networking
- Verifying VSR Installation

9.2 — Verifying Host Details

Successful installation of a VSR VM requires the host machine to be set up properly. Use the commands described in this section to display host information for Linux systems (running Centos or Red Hat).

General System Information

To display the Linux kernel version, enter the following:

```
uname -a ↵
```

To verify that your Linux kernel is 64-bit, enter the following:

```
uname -m ↵
```

The command output should be x86_64.

Linux Distribution Type

To display the type of Linux distribution and version, enter the following:

```
lsb_release -a ↵
```

Note: Depending on your Linux distribution, you may have to install a package such as **redhat-lsb-core** to run this command.

PCI Devices

To view all PCI devices, enter the following:

```
lspci ↵
```

A partial sample output of the command is as follows:

```
[user@host ~]# lspci
04:00.0 Ethernet controller: Intel Corporation 82574L Gigabit Network Connection
05:00.0 Ethernet controller: Intel Corporation 82574L Gigabit Network Connection
06:00.0 Ethernet controller: Intel Corporation 82574L Gigabit Network Connection
07:00.0 Ethernet controller: Intel Corporation 82574L Gigabit Network Connection
```

The first entry indicates that there is a PCI device attached to bus 04, with device ID 00 and function 0 (04:00.0) and that it is an 82574L Gigabit Ethernet controller made by Intel Corporation.

To view PCI device details, including capabilities such as the maximum bus speed and the number of lanes (for example x4), enter the following:

```
lspci -vvv ↵
```

Processor/CPU Information

To view details about all the CPU processors available to the host, enter the following:

```
cat /proc/cpuinfo ↵
```

When hyper-threading is enabled on Intel CPUs, every hyper-thread appears as a separate processor, as shown in the following partial sample output:

```
[user@host ~]# cat /proc/cpuinfo
processor       : 0
vendor_id       : GenuineIntel
cpu family      : 6
model           : 62
model name      : Intel(R) Xeon(R) CPU E5-2630 v2 @ 2.60GHz
stepping        : 4
cpu MHz         : 2593.614
cache size      : 15360 KB
physical id     : 0
siblings        : 12
core id         : 0
cpu cores       : 6
apicid          : 0
initial apicid  : 0
fpu             : yes
fpu_exception   : yes
cpuid level     : 13
wp              : yes
flags          : fpu vme de pse tsc msr pae mce cx8 apic mtrr pge mca cmov pat pse36
clflush dts acpi mmx fxsr sse sse2 ss ht tm pbe syscall nx pdpe1gb rdtscp lm
constant_tsc arch_perfmon pebs bts rep_good xtopology nonstop_tsc aperfmperf pni
pclmulqdq dtes64 monitor ds_cpl vmx smx est tm2 ssse3 cx16 xtpr pdcm dca sse4_1 sse4_2
x2apic popcnt aes xsave avx f16c rdrand lahf_lm ida arat epb xsaveopt pln pts dts
tpr_shadow vnmi flexpriority ept vpid fsgsbase smep erms
bogomips        : 5187.22
clflush size    : 64
cache_alignment : 64
address sizes   : 46 bits physical, 48 bits virtual
power management:
```

To run VSR VMs, the **cpu family** value must be 6 (Intel) and the **model** should be greater than or equal to 42 (in most cases). In addition, several CPU flags are critical for the VSR and must be passed through to the guest. These include:

- x2apic—support for an advanced interrupt controller introduced with Intel Nehalem processors

  Support for the x2apic is mandatory; the flag must not be emulated
- lm—long mode, indicating a 64-bit CPU, which is necessary to support 64-bit guests
- vmx—support for Intel virtualization technologies such as VT-d/VT-x

Host Memory

To view details about the host memory, enter the following:

```
cat /proc/meminfo ↵
```

The following is a partial sample output of this command:

```
[user@host ~]# cat /proc/meminfo
MemTotal:        16406144 kB
MemFree:          9442676 kB
MemAvailable:    11272708 kB
Buffers:           648220 kB
Cached:           1352744 kB
SwapCached:             0 kB
Active:           4898888 kB
Inactive:         1723664 kB
```

The **MemFree** value must be at least 4194304 kB if you want to create another VSR VM on this host.

Host Capability

To show summary information about the host and its virtualization capabilities, enter the following:

```
virsh capabilities ↵
```

Note: The **libvirt** package must be installed to run this command on the host.

The command output must confirm that the system is capable of supporting guests with the x86_64 architecture (64-bit guests).

QEMU and libvirt Information

To show **libvirt** and QEMU version information, enter the following:

```
virsh version ↵
```

Note: The **libvirt** package must be installed to run this command on the host.

The command output must show that the system is capable of supporting guests with the x86_64 architecture (64-bit guests).

Loaded Modules

To list all the kernel modules installed on the host, enter the following:

`lsmod ↵`

Some key modules are: **bridge**, **kvm**, **kvm_intel**, **vhost_net**, **tun**, **macvtap** and **openvswitch**.

Host Virtualization Setup

To check that dependencies for virtualization are installed correctly on the host, enter the following:

`virt-host-validate ↵`

The following is a sample output of the command:

```
[user@host ~]# virt-host-validate
  QEMU: Checking for hardware virtualization                                 : PASS
  QEMU: Checking for device /dev/kvm                                         : PASS
  QEMU: Checking for device /dev/vhost-net                                   : PASS
  QEMU: Checking for device /dev/net/tun                                     : PASS
   LXC: Checking for Linux >= 2.6.26                                         : PASS
[user@host ~]#
```

9.3 — Verifying the Creation of VMs

Before attempting to log in to a VSR system and to check for successful boot of its VMs, ensure that the VMs have been created as expected on all the host machines.

If you are using **libvirt**, you can view the list of VMs on a specific host by entering the following command:

```
virsh list ↵
```

The following is a sample output of this command:

```
[user@host ~]# virsh list --all
 Id    Name                           State
----------------------------------------------------
 1     CPMA                           running
 2     CPMB                           running
 3     IOM1                           running
```

Because each KVM/QEMU virtual machine is a process with two or more threads, you can also use a sequence of commands, such as the following, to get more details about a running VM:

```
[user@host ~]# ps -ef | grep CPMA
qemu     6304    1  5 Sep10 ?       05:03:50 /usr/libexec/qemu-kvm.real -name CPMA
-S -machine rhel6.0.0, accel=kvm, usb=off -cpu SandyBridge, +erms, +smep, +fsgsbase,
+rdrand, +f16c, +osxsave, +pcid, +pdcm, +xtpr, +tm2, +est, +smx, +vmx, +ds_cpl,
+monitor, +dtes64, +pbe, +tm, +ht, +ss, + acpi, + ds, +vme -m 6144 -realtime mlock=off
-smp 2, sockets=2, cores=1, threads=1 -uuid nnnnnnnn-nnnn-nnnn-nnnn-nnnnnnnnnnnn -
nographic -no-user-config -nodefaults -chardev socket, id=charmonitor, path=/var/
lib/libvirt/qemu/CPMA.monitor, server, nowait -mon chardev=charmonitor, id=monitor,
mode=control -rtc base=utc -no-kvm-pit-reinjection -no-shutdown -no-acpi -boot
strict=on -device piix3-usb-uhci, id=usb, bus=pci.0, addr=0x1.0x2 -device virtio-
serial-pci, id=virtio-serial0, bus=pci.0, addr=0x7 -drive file=/path/disk1.qcow2,
if=none, id=drive-virtio-disk0, format=qcow2, cache=none -device virtio-blk-pci,
scsi=off, bus=pci.0, addr=0x8, drive=drive-virtio-disk0, id=virtio-disk0 -netdev
tap, fd=23, id=hostnet0, vhost=on, vhostfd=24 -device virtio-net-pci,
netdev=hostnet0, id=net0, mac=nn:nn:nn:nn:nn:nn, bus=pci.0, addr=0x3, bootindex=1 -
netdev tap, fd=25, id=hostnet1, vhost=on, vhostfd=26 -device virtio-net-pci,
netdev=hostnet1, id=net1, mac=nn:nn:nn:nn:nn:nn, bus=pci.0, addr=0x4 -chardev
socket, id=charconsole0, host=0.0.0.0, port=2500, telnet, server, nowait -device
virtconsole, chardev=charconsole0, id=console0 -device virtio-balloon-pci,
id=balloon0, bus=pci.0, addr=0x6


[user@host ~]# ps -T 6304
root@bksim4204 6304]# ps -T 6304
  PID  SPID TTY      STAT   TIME COMMAND
 6304  6304 ?        Sl     0:19 /usr/libexec/qemu-kvm.real -name CPMA -S -machine
rhel6.0.0,accel=k
 6304  6310 ?        Sl   169:47 /usr/libexec/qemu-kvm.real -name CPMA -S -machine
rhel6.0.0,accel=k
 6304  6311 ?        Sl   134:52 /usr/libexec/qemu-kvm.real -name CPMA -S -machine
rhel6.0.0,accel=k
```

These sample command outputs indicate that the VM called CPMA is running as process ID 6304 in the host machine. There are three threads associated with this process.

You can obtain a real-time view of the host system impact of all running VMs. by entering the following commands;

```
top ↵
```

```
htop ↵
```

The following is a sample output of the command:

```
[root@bksim4204 bin]# top
top - 14:00:10 up 5 days,  4:44,  2 users,  load average: 4.09, 4.08, 4.09
Tasks: 184 total,   2 running, 182 sleeping,   0 stopped,   0 zombie
%Cpu(s): 51.2 us,  0.1 sy,  0.0 ni, 48.7 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0 st
KiB Mem:  16406144 total,  6970448 used,  9435696 free,   649488 buffers
KiB Swap:        0 total,        0 used,        0 free.  1328612 cached Mem

  PID USER     PR  NI    VIRT    RES    SHR S  %CPU %MEM     TIME+ COMMAND
 6349 qemu     20   0 4860828 2.445g  5912 S 402.1 15.6  24522:22 qemu-kvm.real
 6304 qemu     20   0 6736452 0.981g  5916 S   4.7  6.3 305:44.16 qemu-kvm.real
 6321 qemu     20   0 6736424 0.980g  5916 S   4.3  6.3 280:32.31 qemu-kvm.real
 6308 root     20   0       0      0     0 S   0.3  0.0   4:45.88 vhost-6304
 6324 root     20   0       0      0     0 S   0.3  0.0   1:23.12 vhost-6321
```

9.4 — Verifying Host Networking

You can use different methods to provide network connectivity between the VSR VMs and external destinations. See Chapter 5, VSR Network Connectivity for detailed information.

Table 9.1 lists some useful commands that can help you troubleshoot networking at the host level.

Table 9.1 — Host Network Troubleshooting

Command Syntax	Description
ip -d link show	Shows details of all host network interfaces, including physical NIC ports and logical interfaces, such as vNIC interface constructs on the host
ip link set dev <interface-name> **mtu** <value>	Explicitly sets the MTU (Maximum Transmit Unit) of a host interface Alcatel-Lucent strongly recommends that you set the MTU of network interfaces associated with distributed model VSR-SIM internal fabric ports to 9000 bytes If this is not practical and the maximum possible value is N bytes, then you should lower the MTU of all IOM ports to less than or equal to N-64 bytes
ip addr show	Displays the IP addresses associated with host network interfaces **Note**: In a virtualization environment, many interfaces will not have any IP addresses assigned to them
ip route show	Displays the IP routing table of the host
tcpdump -i <interface name>	Captures packets on the selected interface and outputs them for analysis
brctl show	Displays all the Linux bridges
ovs-vsctl show	Displays all the Open vSwitch bridges
ethtool -S <interface name>	Displays statistics collected by the physical NIC for a selected interface

9.5 — Verifying VSR Installation

Once you have verified that the VSR VMs have been created successfully on the respective hosts, check the SR OS operating system to verify that it has booted up properly on each VM and that each VSR system is functional. You will typically need console access to the VSR guests to perform these checks. This is the main reason for adding a serial console port to VSR VMs. With console access, you can log in to each VSR and perform the checks that are described in this section.

Check the Status of the System BOF

The output of this check depends on the SMBIOS text string you used for the VM and the saved BOF configuration. At the prompt, type the following:

```
A:vSIM#  show bof ↵
```

The following is a sample output of this command:

```
===========================================================
BOF (Memory)
===========================================================
primary-imagecf3:\timos\both.tim
primary-configcf3:\config.cfg
license-filecf3:\timos.alu.a7.txt
address1.1.1.1/24 active
static-route  1.0.0.0/8
next-hop 192.0.2.1
autonegotiate
duplexfull
speed100
wait3
persistoff
no li-local-save
no li-separate
console-speed115200
cf3:\timos.alu.a7.txt
===========================================================
```

Check that the Chassis Type is Set Correctly

If the VSR VM is set up as a VSR-RR, the chassis type displays as 7750 SR-12 and you cannot modify the chassis type.

Note: The **Chassis Topology** field is used to differentiate an XRS-20 from an XRS-40, not to differentiate an integrated model VSR-SIM from a distributed model VSR-SIM.

If the chassis type does not match the one encoded in the SMBIOS text string, you should assume there is an error in the SMBIOS text string. To view the chassis information, type the following at the prompt:

A:vSIM# show chassis ↵

The following is a sample output of this command:

```
===============================================================================
System Information
===============================================================================
Name                               : vSIM
Type                               : 7750 SR-c4
Chassis Topology                   : Standalone
Location                           : (Not Specified)
Coordinates                        : (Not Specified)
CLLI code                          :
Number of slots                    : 2
Oper number of slots               : 2
Number of ports                    : 0
Critical LED state                 : Off
Major LED state                    : Off
Minor LED state                    : Off
Over Temperature state             : OK
Base MAC address                   : 30:18:ff:00:00:01
===============================================================================
Chassis Summary
===============================================================================
Chassis       Role         Status
-------------------------------------------------------------------------------
1             Standalone   up
===============================================================================
```

Check that the Correct (Virtualized) Card Types are Equipped in the System

If the VSR VM is set up as a VSR-RR, the card in slot A displays as an sfm4-12 and the card in slot 1 displays as an iom3-xp-b; you cannot modify these card types.

If a card type does not match the one encoded in the SMBIOS text string of the corresponding VM, you should assume there is an error in that SMBIOS text string. To view the card information, type the following at the prompt:

A:vSIM# show card ↵

The following is a sample output of this command:

```
===============================================================================
Card Summary
===============================================================================
Slot      Provisioned Type                         Admin Operational   Comments
              Equipped Type (if different)         State State
-------------------------------------------------------------------------------
1         xcm-x20                                  up    up
A         cpm-x20                                  up    up/active
B         cpm-x20                                  up    up/standby
===============================================================================
```

Check that the VSR system has valid licenses

To view the system license information, type the following at the prompt:

`A:vSIM# show system license ↵`

The following is a sample of the command output:

```
A:vsr-sim# show system license

===============================================================================
System  License
===============================================================================
License status : monitoring, valid license record
Time remaining : 178 days 5 hours
===============================================================================

===============================================================================
Active  License [CPM A]
===============================================================================
License status : monitoring, valid license record
Time remaining : 178 days 5 hours
-------------------------------------------------------------------------------
License name   : network-operator.com
License uuid   : 8664885f-db82-4f0f-8f5f-0ba15275ae91
Machine uuid   : 8664885f-db82-4f0f-8f5f-0ba15275ae91
License desc   : joe@network-operator.com
License prod   : Virtual-SIM
License sros   : TiMOS-[BC]-13.0.*
Current date   : FRI SEP 04 18:55:16 UTC 2015
Issue   date   : MON AUG 24 22:48:41 UTC 2015
Start   date   : TUE SEP 01 00:00:00 UTC 2015
End     date   : TUE MAR 01 00:00:00 UTC 2016
===============================================================================

===============================================================================
Standby License [CPM B]
===============================================================================
License status : monitoring, valid license record
Time remaining : 178 days 5 hours
-------------------------------------------------------------------------------
License name   : network-operator.com
License uuid   : 6ef11a9c-d8ad-4912-8842-069c19cc37a7
Machine uuid   : 6ef11a9c-d8ad-4912-8842-069c19cc37a7
License desc   : joe@network-operator.com
License prod   : Virtual-SIM
License sros   : TiMOS-[BC]-13.0.*
Current date   : FRI SEP 04 18:55:17 UTC 2015
Issue   date   : MON AUG 24 22:48:41 UTC 2015
Start   date   : TUE SEP 01 00:00:00 UTC 2015
End     date   : TUE MAR 01 00:00:00 UTC 2016
===============================================================================
```

Customer documentation and product support

Customer documentation

http://documentation.alcatel-lucent.com

Technical support

http://support.alcatel-lucent.com

Documentation feedback

documentation.feedback@alcatel-lucent.com

3HE 10848 AAAA TQZZA Edition 01