Oracle® Communications Session Border Controller and Session Router

VNF Essentials Guide

Release S-CZ7.3.9

March 2018

ORACLE

Oracle Communications Session Border Controller and Session Router VNF Essentials Guide, Release S-CZ7.3.9

Contents

B DPDK-Based OVS Installation

List of Figures

About this Guide

Oracle® Communications Session Border Controller (OCSBC) applies core session control to reduce the complexity and cost of delivering high-value, revenue generating SIP multimedia services. Oracle Communications Session Border Controller can be used to support a broad range of SIP services including residential or business voice, GSMA-defined Rich Communication Suite (RCS) services and fixed mobile convergence (FMC) for small subscriber populations or initial service rollouts.

Oracle Communications Session Router (OCSR) is a next generation session routing proxy that incorporates both IETF SIP proxy and 3GPP Breakout Gateway Control Functions (BGCF). It is specifically design to control the routing of large volumes of SIP session signaling messages. Session Router provides high-performance SIP routing with scalable routing policies that increase overall network capacity and reduce costs. It plays a central role in Oracle's open session routing (OSR) architecture and helps service providers build a scalable, next-generation signaling core for SIP-based services.

Both the Oracle Communications Session Border Controller (OCSBC) and Oracle Session Router (OCSR) are capable of being deployed as Virtual Network Functions (VNFs). Support for the OCSBC and OCSR as VNFs is the same. This document refers to the OCSBC, but the reader can assume that all information herein applies equally to the OCSR.

Release Version S-Cz7.3.9 is supplied as virtual machine software or as a software-only delivery suitable for operation on server hardware. Refer to sales documentation updates for information further specifying hardware support.

Release Feature Support

To obtain the latest feature information for this release, please contact your Oracle sales representative.

Performance Information

Please refer to your Oracle Sales Consultant for available documentation on product performance based on Oracle's testing results.

Related Documentation

The following table lists the members that comprise the S-Cz7.3.0 documentation set, to which this release is related:

Document Name	Document Description
Release Notes	Contains information about the current documentation set release, including new features and management changes.
ACLI Configuration Guide	Contains information about the administration and software configuration of the Oracle Communications Session Border Controller.

Document Name	Document Description
ACLI Reference Guide	Contains explanations of how to use the ACLI, with alphabetical listings and descriptions of all ACLI commands and configuration parameters.
Maintenance and Troubleshooting Guide	Contains information about logs, performance announcements, system management, inventory management, upgrades, working with configurations, and managing backups and archives.
MIB Reference Guide	Contains information about Management Information Base (MIBs), Acme Packet's enterprise MIBs, general trap information, including specific details about standard traps and enterprise traps, Simple Network Management Protocol (SNMP) GET query information (including standard and enterprise SNMP GET query names, object identifier names and numbers, and descriptions), examples of scalar and table objects.
Accounting Guide	Contains information about accounting support, including details about RADIUS and Rf accounting.
HDR Resource Guide	Contains information about the Historical Data Recording (HDR) feature. This guide includes HDR configuration and system-wide statistical information.
Administrative Security Essentials	Contains information about support for the Administrative Security license.
Security Guide	Contains information about security considerations and best practices from a network and application security perspective for the Oracle Communications Session Border Controller family of products.

Training

Oracle offers extensive training on products via Oracle University. Find information on classes, curriculum and registration at http://education.oracle.com. Instructor-led, remote instructor, and self-directed classes are available, depending on the subject. Curriculum can include lecture and hands-on activities.

The *Oracle SBC Configuration and Administration* class provides an introduction to the Session Border Controller.

Support and Advanced Customer Services

Oracle provides access to product documentation, service request tools, and other related information via My Oracle Support. Oracle also offers expert Advanced Customer services on this and all other products via My Oracle Support. Information about service offerings and procurement is available at https://support.oracle.com.

Intended Audience

The information presented in this document is intended for Sales Engineers, Solution Architects, Implementation Engineers, and Support Technicians working on NFV

environments. This document explains Oracle's solution and its configuration, meeting customer needs at a high level.

Revision History

Date	Description
March 2017	• Initial Release
March 2017	• Removes incorrect reference to SLB
May 2017	• Corrects typo in 'About this Guide' • Adds caveat on unsupported GUI
June 2017	• Removes IMS-AKA/RTP caveat
December 2017	• Adds Upgrade Caveat about updating companion bootloader for p4+ upgrade
March 2018	• Adds VMAC procedure for VNF HA

1

The Oracle Communications SBC and SR as VNFs

The OCSBC and OCSR are capable of being deployed as VNFs. Support for the OCSBC and OCSR as VNFs is the same. This document refers to the OCSBC, but the reader can assume that the information applies equally to the OCSR.

VNF deployment types include:

- A standalone (not orchestrated) instance Oracle Communications Session Border Controller operating as a virtual machine running on a hypervisor, and
- Virtual Machine(s) deployed within an Orchestrated Network Functions Virtualization (NFV) environment.

Standalone OCSBC VNF deployment instances are always supported. Support within an orchestrated environment is dependent on orchestrator and OCSBC version. High Availability configurations are supported by both deployment types.

Qualified Hypervisors, Hardware and I/O Modes

For version S-Cz7.3.9, Oracle has qualified the components listed in this section.

Oracle has qualified this version of the Oracle Communications Session Border Controller to run on the following hypervisors:

- XEN 4.4: Specifically using Oracle Virtual Machine (OVM) 3.4.2
- KVM: Using version embedded in Oracle Linux 7 with UEK4u1.
  Note the use of the following KVM component versions:
  - Compiled against library: libvirt 2.0.0
  - Using library: libvirt 2.0.0
  - Using API: QEMU 2.0.0
  - Running hypervisor: QEMU 1.5.3
- VMware: Using ESXI 6.0 u2 on VMware vCenter Server

Qualified hardware platforms include:

- Netra X5-2
- Oracle X5-2

Qualified interface chipsets include:

- Intel x540/82599
- Intel i350
- Intel X710 / XL710

Qualified interface I/O modes include:

- PV (VIF on XEN)
- SR-IOV
- PCI Passthrough

Virtual Machine Platform Resources

An OCSBC and OCSR virtual machine requires CPU core, memory, disk size, and network interfaces specified for operation. The system uses the Data Plane Development Kit (DPDK) for datapath design, which imposes VNF-specific resource requirements for CPU cores. Deployment details, such as the use of distributed DoS protection, specify resource utilization beyond the defaults.

The user configures CPU core utilization from the ACLI based on their deployment. The user can also define memory and hard disk utilization based on deployment, but must configure the hypervisor with these settings prior to startup if they need them to be different then the machine defaults setup by the machine template (OVA).

Default VM Resources

VM resource configuration defaults to the following:

- 4 CPU Cores - specific to deployment (See *Core Configuration* in this document)
- 8 GB RAM
- 40 GB hard disk (pre-formatted)
- 8 interfaces as follows:
  - 1 for management (wancom0)
  - 2 for HA (wancom1 and 2)
  - 1 spare
  - 4 for media

Interface Host Mode

The Oracle Communications Session Border Controller S-CZ7.3.9 VNF supports interface architectures using Hardware Virtualization Mode - Paravirtualized (HVM-PV):

- ESXi - No manual configuration required.
- KVM - HVM mode is enabled by default. Specifying PV as the interface type results in HVM plus PV.
- XEN (OVM) - The user must configure HVM+PV mode.

CPU Core Resources

The Oracle Communications Session Border Controller S-CZ7.3.9 VNF requires an Intel Core2 processor or higher, or a fully emulated equivalent including 64-bit SSSE3 and TSC support.

If the hypervisor uses CPU emulation (qemu etc), Oracle recommends that the deployment passes the full set of host CPU features to the VM.

Hyper-threading and CPU Affinity

Due to the polling operation of DPDK, using Hyper-threaded cores can degrade the OCSBC VNF's performance. Oracle recommends the user disable Hyper-threading in the host system if possible, or configure CPU affinities on the hypervisor to ensure mapping from only one virtual CPU to each physical CPU core.

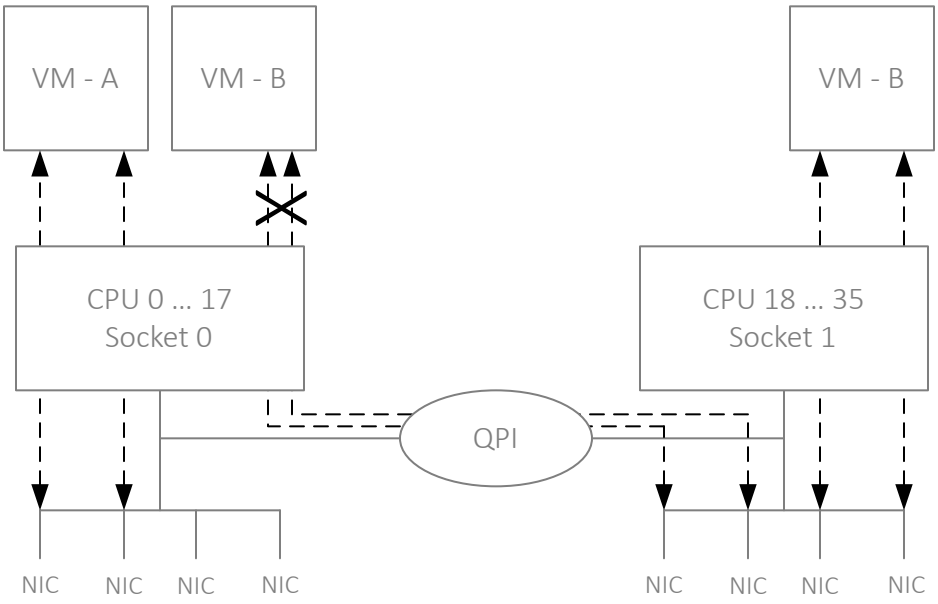
The user can learn how to configure CPU affinity via their hypervisor documentation.

Host Hypervisor CPU Pinning

Many hardware platforms have built in optimizations related to VM placement. For example, some CPU sockets may have faster local access to Peripheral Component Interconnect (PCI) resources than other CPU sockets. Users should ensure that VMs requiring high media throughput are optimally placed by the hypervisor, so that traversal of cross-domain bridges, such as QuickPath Interconnect (QPI), is avoided or minimized.

Some hypervisors implement Non-Uniform Memory Access (NUMA) topology rules to automatically enforce such placements. All hypervisors provide manual methods to perform CPU pinning, achieving the same result.

The diagram below displays two paths between the system's NICs and VM-B. Without configuring pinning, VM-B runs on Socket 0, and has to traverse the QPI to access Socket 1's NICs. The preferred path pins VM-B to Socket 1, for direct access to the local NICs, avoiding the QPI.



The user can learn how to pin CPUs via their hypervisor documentation.

Distribution Components

The Oracle Communications Session Border Controller VNF version S-CZ7.3.9 is a release suited for deployment by a virtual machine manager. Oracle provides virtual machine templates specific to the hypervisor over which the product is deployed.

Oracle provides the version S-CZ7.3.9 Oracle Communications Session Border Controller VNF the following distributions:

- `nnSCZ739-img-vm_ovm.ova`—Open Virtualization Archive (.ova) distribution of the Oracle Communications Session Border Controller VNF for Oracle (XEN) virtual machines.
- `nnSCZ739-img-vm_kvm.tgz`—Compressed image file including Oracle Communications Session Border Controller VNF for KVM virtual machines.
- `nnSCZ739-img-vm_vmware.ova`—Open Virtualization Archive (.ova) distribution of the Oracle Communications Session Border Controller VNF for ESXi virtual machines.

The Oracle (XEN) Virtual Machine, KVM, and ESXi packages include:

- Product software—Bootable image of the product allowing startup and operation as a virtual machine. This disk image is either in the vmdk or qcow2 format.
- `usbc.ovf`—XML descriptor information containing metadata for the overall package, including identification, and default virtual machine resource requirements. The .ovf file format is specific to the supported hypervisor.
- `legal.txt`—Licensing information, including the Oracle End-User license agreement (EULA) terms covering use of this software, and third-party license notifications.

Configuration Overview

Oracle Communications Session Border Controller VNF deployments require configuration of the virtual machine environment and of the SBC itself. The user can consider SBC configuration as separate from VNF configuration. VNF-specific configuration allows for resource tuning, such as CPU core allocation, based on the deployment's performance and capacity requirements.

When the user deploys the SBC VNF, they can configure operational information within the system's boot parameters, such as:

- Fixed management interface IP address
- Host name

In addition, the user must perform basic SBC configuration procedures after installation, including:

- Setting Passwords
- Setup Product
- Setup Entitlements

The user can refer to the *ACLI Configuration Guide* for instructions on these procedures.

For Xen-based hypervisors, the default boot mode uses DHCP to obtain an IP address for the first management interface (wancom0) unless a static IP is provisioned. Note that DHCP on wancom0 does not support lease expiry, so the hypervisor must provide persistent IP address mapping. If persistent IP address mapping is not provided, the user must manually restart the VM whenever the wancom0 IP address changes due to a manual change or DHCP lease expiry.

Alternatively, your virtual machine or orchestration tools may provide a means of configuring operational information.

Performance and capacity default values are configured for a typical VNF deployment. Application parameters can be further optimized by the user after deployment, including:

- Media manager traffic/bandwidth utilization tuning
- Datapath-related CPU core allocation

See the section on *Configuring Your Device for NFV* for VNF tuning configuration. Refer to your VM documentation for VM environment configuration. Refer to the documentation listed in the *About This Guide* section of this document for SBC-related configuration.

Co-Product Support

The products/features listed in this section run in concert with the Oracle Communications Session Border Controller for their respective solutions.

Oracle Communications Session Delivery Manager

Oracle Communications Session Delivery Manager 7.5M3 and later support this GA release of the Oracle Communications Session Border Controller

Oracle Communications Application Orchestrator

Oracle Communications Application Orchestrator 1.1M3 and later support this GA release of the Oracle Communications Session Border Controller

Oracle Communications Session Border Controller

Oracle Communications Session Border Controller versions 7.3.0 and later support this GA release for pooled transcoding.

Transcoding Support

The system performs transcoding using embedded DSPs (TCU) or an external transcoder. The user configures the VNF to perform embedded transcoding by configuring it with one or more transcoding cores. The user configures pooled transcoding to utilize an external transcoding SBC.

This version of the Oracle Communications Session Border Controller supports embedded, software-based transcoding of and between the following codecs:

- G.711 PCMU/PCMA
- G.729 A/B

Refer to the *ACLI Configuration Guide* for Command Line Interface instructions on configuring transcoding. References in the *ACLI Configuration Guide* to hardware-based operation is not relevant to embedded, software-based transcoding.

VLAN Support

This version of the OCSBC supports VLANs within all supported virtual machine environments and interface architectures. Refer to the *ACLI Configuration Guide* for instructions on configuring VLANs on the OCSBC. Note that when VLANs are configured, the OCSBC requires VLAN tags to be included in the packets delivered to/from the VM.

The user should evaluate the VLAN support of their deployment's hypervisor and interface I/O mode before implementation to ensure secure support for the transmission and receiving of VLAN-tagged traffic. Please consult your hypervisor’s vendor documentation for details.

Provisioning Entitlements

VNF products licensing follows the standard C-series self-entitlements licensing model. Refer to the *ACLI Configuration Guide* for instructions on setting entitlements.

SPL and Baseline Information

SPL Version Support

Current SPL Engine version supported:

- C2.0.0
- C2.0.1
- C2.0.2
- C2.0.9
- C2.1.0
- C2.1.1
- C2.2.0
- C2.2.1
- C2.3.2
- C3.0.0
- C3.0.1
- C3.0.2
- C3.0.3
- C3.0.4
- C3.0.6
- C3.0.7
- C3.1.0
- C3.1.1
- C3.1.2
- C3.1.3
- C3.1.4
- C3.1.5
- C3.1.6

Release and Patch Baseline

Current Patch Baseline: SCZ7.3.0m2p2 and 729p6.

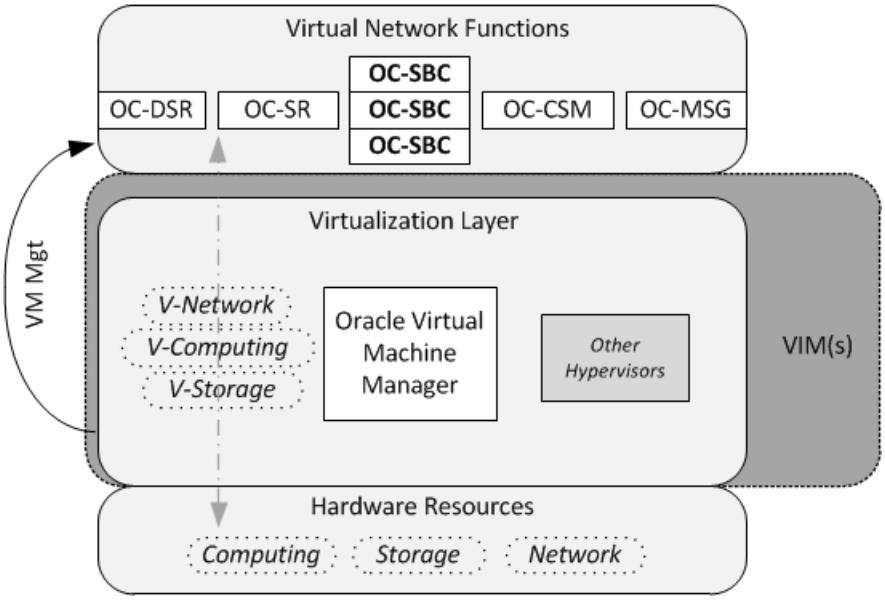
2
Network Functions Virtualization Overview

Network Functions Virtualization (NFV) is a set of architectural standards to leverage enterprise and cloud technologies for virtualization and lifecycle resource automation in telecommunications core networks. It is focused on identifying the use cases and needs of communications service providers for automated, virtualized infrastructure and network functions, along with an architecture to frame discussions on meeting those needs, and an analysis of the gaps with existing technologies. Standards defining the NFV architecture addressed in this document include "ETSI GS NFV 002 Architectural Framework", ETSI GS NFV 004 "Virtualization Requirements" and related. Oracle can provide its OCSBC and OCSR, as well as other network systems, as Virtual Network Functions (VNFs) or Management Systems for operation within an NFV-enabled environment.

One of the main drivers for NFV is capital equipment cost reduction for CSPs. Virtual functions running on fewer COTS servers is much cheaper than using multiple purpose built appliances. NFV meets the needs for cost reduction in addition to allowing the CSP to turn up new services and capacity more rapidly.

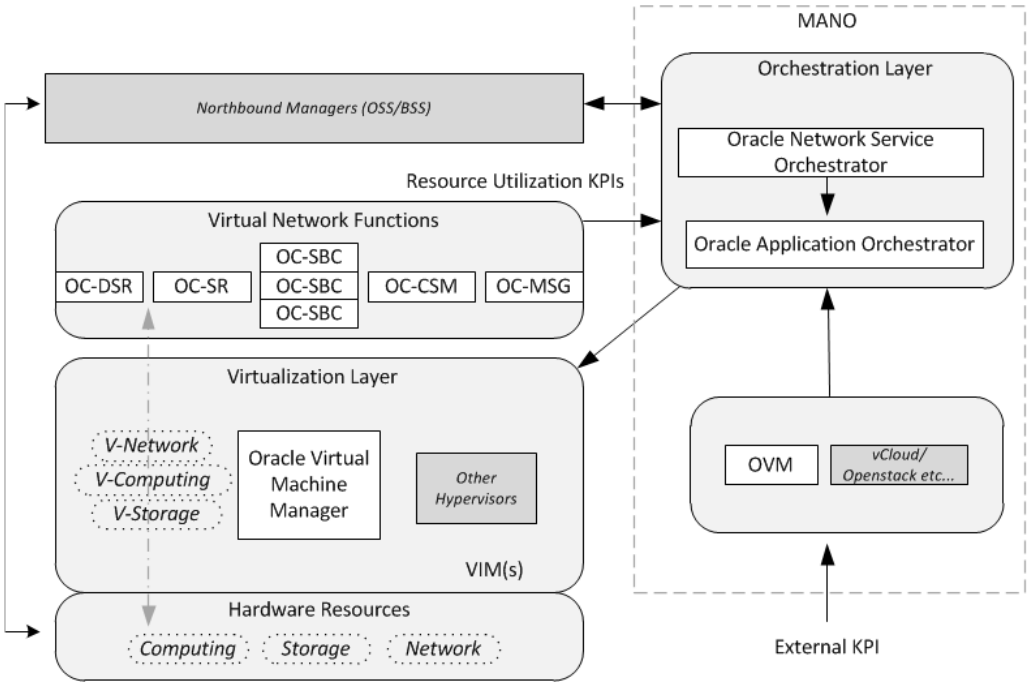
It is generally recognized now that operators desire to move their core network functions into automated and virtualized environments based on these technologies, starting with large operators in North America, Europe, and Asia. To accomplish this requires changes in the infrastructure that hosts the network functions, changes in the network functions, and additions to the management and automation of the life cycle of network services.

In the NFV Industry Specification Group under ETSI, which is where the bulk of industry focus on defining the use cases and architecture for NFV have been, an architectural framework, shown below, has been defined that is the basis for most discussions of the various functional components needed for an NFV environment. These components are divided into the Hardware, the VNFs and the virtualization layer. Commodity hardware provides the physical components for computing, and are the same as those without virtualization. The virtualization layer provides the means through which virtual machines can operate. The VNFs are the virtual machines themselves, instantiated as Virtual Network Functions (VNFs). These components establish a VNF deployment that can be fully operational, but without orchestration.



Extending upon these basic operational components are the Management and Orchestration components (MANO) components. MANO provides a means by which operators approach a zero-touch orchestration, operation and management deployment. The resulting automated, elastic construct more fully addresses the OPEX efficiencies and real-time responsiveness desired from NFV.

To complete the NFV picture, MANO components are included in the diagram below, consisting of the orchestration layer and VIM components associated with managing your NFV. This diagram also adds the northbound OSS/BSS systems. The orchestration layer supports NFV instantiating and de-commissioning based on operational requirements and status, as well as on business requirements and preferences. The MANO model also provides for basic FCAPS management, encapsulating those management functions outside of orchestration.



Network operators establish criteria for automated VNF instantiation within the orchestration layer, by configuring the overall system to use external and internal status within KPIs to define

the triggers for changes to VNF resources. VNFs themselves provide critical status information that, when combined with organizational policy and other status triggers, the MANO uses to adjust network resources to network requirements in real time.

Oracle's NFV solution harnesses the extensive assets from its portfolio to deliver a vision of a network orchestration being driven by a business orchestration, which in turn delivers business value to the customer. This vision builds on the investment that Oracle Communications has made in the BSS/OSS domain to deliver a BSS/OSS that is orchestrated along business processes. The Oracle Communication Network Service Orchestrator provides the join of the two worlds by bridging the chiasm between Network Orchestration and Business Orchestration.

Oracle's NFV solution builds on the investment Oracle has made in the network domain to have an extensive portfolio of Network Functions as well as network orchestration.

3

Introduction to Platform Preparation and Software Deployment

This section introduces the deployment of Oracle Communications software onto the platforms supported by this version of the Oracle Communications Session Border Controller. if applicable, refer to Hardware Installation documentation for rack and stack procedures. This documentation explains platform preparation, power-on and software deployment to the point where the user is ready to perform service configuration.

Platform support by Oracle Communications session delivery software extends to generic platforms. As such, preparation of these platforms requires that the user perform tasks independent of their session delivery product. Although the user needs to use platform documentation for this platform-specific information, Oracle distributes this documentation to provide the user with:

- Settings required by the session delivery software.
- Guidance to procedures that apply to session delivery software.

Virtual Machines

Virtual Machines (VMs) supported by Oracle Communications Session Delivery software varies across software version. Find specific version support within the context of your version's documentation.

Operation over VMs is roughly equivalent to deployment over COTS/Server hardware. Platform preparation, however, differs greatly. In addition, platform preparation differs greatly between VM platforms.

Preparation procedures that apply to all VM platforms include the following steps:

1. Make the VM template available to the VM manager.
2. Configure the VM manager to apply the template correctly for Oracle Communications Session Delivery software.
3. Power-on the VM. If the deployment is using a VM template, the system uses that template to automatically install onto the virtual drive, after which the server reboots. Deployments using raw images do not perform an installation process.

VM deployment requires extensive knowledge about the specific platform that is not documented herein. The intent of this documentation is to provide information that helps the user navigate the deployment and perform tasks that are specifically related to Oracle Communications Session Delivery software.

4
Virtual Machine Platforms

Oracle distributes virtual machine templates, each containing a virtual disk image and default configuration for the supported profile of each VM platform. VM platform support is dependent on your Oracle product version.

This section addresses requirements associated with running applicable software as virtual machines. It also provides basic instructions on loading and starting machine templates.

VM distributors maintain extensive documentation sites. You must use those vendors' documentation for full explanations and instructions on VM deployment and operation.

Create and Deploy on Oracle VM Manager

This section provides detail on deploying Oracle Communications Session Delivery products in an Oracle Virtual Machine environment and assumes Oracle VM Manager 3.4.2. The section also assumes the user understands deployment in these environments and that the majority of deployment tasks, from hardware installation and startup to VM resource and management setup, is complete.

For information on Oracle OVM, including server and manager documentation, refer to the following links. The bottom link opens the page specifically to Oracle OVM version 3.4.2.

http://www.oracle.com/technetwork/documentation/vm-096300.html
http://docs.oracle.com/cd/E64076_01/

Once hardware, VM resources and VM management are in place, the user prepares the VM management environment. High level steps include:

- Discover Oracle VM Servers
- Discover Storage (File Server or SAN)
- Create a Virtual Machine Network
- Create a Server Pool
- Create a Storage Repository

> **Note:**
>
> The following procedure describes a typical deployment. The system may display different screens, depending on the deployment.

Oracle Communications Session Delivery product-specific setup steps include the following.

- Add Resources to Storage Repository
  - Import an assembly
  - Import a virtual machine template (from the assembly)

- Create a virtual machine from a template
- Configure processing resources
- Assign Networks
- Specify Boot Order
- Connect to the console
- Start your virtual machine

Use the Oracle VM Manager to deploy your VMs. Browsing the manager displays the management application, with tabs across the top for VM Manager configuration and display, and two panes providing access to controls and components for specific VM management.

Follow the steps below to deploy your VM(s):

1. From the Oracle VM Manager application Home page, navigate to the **Repositories** tab.
2. Click the **Virtual Appliances** folder on the left pane, then click the download icon from the center pane.

   Oracle VM Manager displays the **Import Virtual Appliance** dialog.
3. Type the URL, either the HTTP or the FTP server, of your .ova appliance into the **Virtual Appliance download location** text box.
4. Check the **Create VM** checkbox.
5. Select the server pool to which your new machine belongs from the **Server Pool** dropdown.
6. Click OK.

   Oracle VM manager creates your Virtual Machine.
7. Select **VM Files** folder on the left pane and verify the file is present.
8. Select the **Servers and VMs** tab and select the server pool.
9. Click on the **Perspective** drop down menu and select **Virtual Machines**.
10. Right click on the Virtual Machine and select **Edit**.
11. Edit your VM to configure the desired CPU, memory and core resources. Consider the following settings:
    - Operating System - Oracle Linux 7
    - Domain Type - XEN HVM PV Drivers
    - Memory - Set according to your deployment (defaults to 8G)
    - Processors - Set according to your deployment (defaults to 4)
12. Open the **Networks** tab to manage your networks using the **Network** drop-down selection box. OVM does not display MAC addresses until the user applies the configuration.

    The SBC enumerates and binds network interfaces in the order presented by the hypervisor to the virtual machine. If it presents 3 or less interfaces, the bind order is wancom0, s0p0, s1p0. If it presents more than 3 interfaces, the bind order is:

    a. wancom0
    b. wancom1
    c. wancom2
    d. spare

e. s0p0

f. s1p0

g. s0p1

h. s1p1

If your hypervisor randomly allocates addresses for network interfaces, the interface order at the hypervisor does not necessarily match that at the SBC. You can use the **interface-mapping show** command to determine the MAC address to interface order, and if necessary, adjust it using the **interface-mapping swap** command.

13. If you want to increase the default disk size, click the **Disks** tab and the pencil icon to set a larger disk size.

    Do not decrease disk size.

14. Click the **Boot Order** tab and ensure that Disk is the first (or only) option in the boot order list.

15. Click OK.

    The system applies your settings to your VM.

16. Click the **Console** icon from the menu bar.

    Oracle VM Manager displays a terminal screen with the serial CLI operational.

17. Highlight the target and click the **Start** button.

    Oracle VM Manager starts your VM, displaying the startup sequence and, ultimately, providing ACLI access in the console.

Create and Deploy on KVM

For complete KVM documentation, refer to http://www.linux-kvm.org/page/Documents.

1. Install the Virtualization Host group and virt-install.

```
# yum groupinstall "Virtualization Host"
# yum install virt-install
```

2. Extract the image.

```
# tar xvf nnSCZ739.64-img-vm_kvm.tar
nnSCZ739.64-img-vm_kvm.ovf
nnSCZ739.64-img-vm_kvm.qcow2
legal.txt
```

3. Use virt-manager to create the management and media network interfaces.
   - Create a virtual bridged network for management interfaces.
   - Create virtual networks for media interfaces.

4. Provision a new virtual machine.

```
# virt-install \
    --name SBC739 \
    --description "nnSCZ739 KVM" \
    --os-type=Linux \
    --os-variant=rhel7 \
    --ram=8192 \
    --vcpus=4 \
    --disk path=/opt/nnSCZ739.64-img-
```

```
vm_kvm.qcow2,bus=virtio,size=10,format=qcow2 \
    --network bridge=br-Mgmt \
    --network bridge=br-Mgmt \
    --network bridge=br-Mgmt \
    --network bridge=br-Mgmt \
    --network network=media1 \
    --network network=media2 \
    --import \
    --cpu host
```

> **Note:**
>
> Use interface-mapping to pin the four br-Mgmt network interfaces to wancom0, wancom1, wancom2, and spare.

`--name`
Identify a unique name for the virtual machine on this hypervisor.

`--description`
Describe this virtual machine.

`--os-type`
Specify the operating system type.

`--os-variant`
Optimize the configuration for a specific operating system.

`--ram`
Allocate a specific amount of RAM to this virtual machine.

`--vcpus`
Allocate a specific number of virtual CPUs to this virtual machine.

`--disk`
Specify the path to the disk image.

`--network`
Connect the virtual machine to a host network.

`--import`
Skip the operating system installation process and build a guest around the disk image specified with `--disk`.

`--cpu`
Configure the CPU model and CPU features exposed to the virtual machine.

See `man virt-install` for more information.

> **Note:**
>
> The `--cpuset` and `--numatune` options may be used to establish CPU affinity and socket pinning.

Create and Deploy on VMware®

This section provides detail on deploying Oracle Communications Session Delivery products over the ESXI hypervisor and assumes VMware 6. The section also assumes the user understands deployment in these environments and that the majority of deployment tasks, from hardware installation and startup to VM resource and management setup, is complete.

For information on VMware 6, which is also supported, refer to the following link.

https://www.vmware.com/support/pubs/vsphere-esxi-vcenter-server-6-pubs.html

Before You Begin:

- Confirm that the VMware version 6 Hypervisor is installed on an appropriate network server.
- Confirm that the server has 40GB of space for this installation.

> **Note:**
>
> The following procedure describes a typical deployment. The system may display different screens, depending on the deployment.

Detail on Oracle Communications Session Delivery product-specific setup steps is shown below.

1. On the vSphere Client application Home page, go to File > Deploy OVF Template File.
2. On the Source screen, browse to the target .ova file, and click Next.
3. On the End User License Agreement screen, click Accept and click Next.
4. On the Name and Location screen, do the following and click Next.
   - Name. Enter a name for the template.
   - Inventory Location. Select the location where you want to deploy the template.
5. On the Host / Cluster screen, select the host or cluster where you want to run the deployed template, and click Next.
6. If applicable to your deployment, select the resource, from the Resource Pool screen, in which you want to deploy this template, and click Next.
7. On the Storage screen, select the destination storage for the virtual machine files, and click Next.
8. On the Disk Format screen, select Thick Provisioned Lazy Zeroed, and click Next.
9. On the Network Mapping screen, map the networks used in this template to networks in your inventory, and click Next.

   The SBC enumerates and binds network interfaces in the order presented by the hypervisor to the virtual machine. If 3 or less interfaces are presented, the bind order is wancom0, s0p0, s1p0. If more than 3 interfaces are presented, the bind order is:

   a. wancom0
   b. wancom1

c. wancom2

d. spare

e. s0p0

f. s1p0

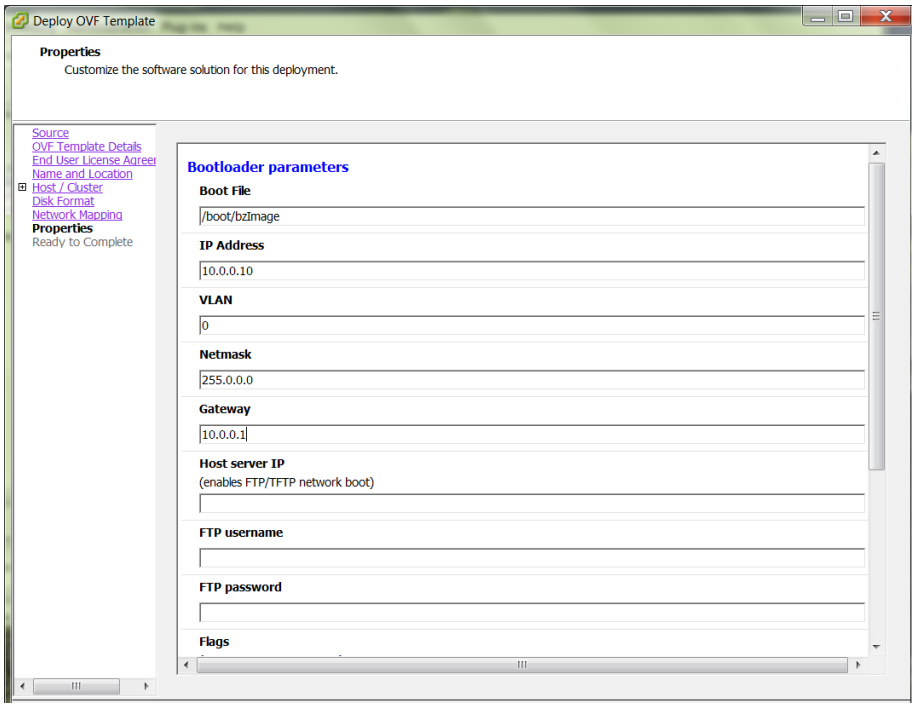
g. s0p1

h. s1p1

If your hypervisor randomly allocates addresses for network interfaces, the interface order at the hypervisor does not necessarily match that at the SBC. You can use the **interface-mapping show** command to determine the MAC address to interface order, and if necessary, adjust it using the **interface-mapping swap** command.

10. On the properties screen, enter parameters per your configuration, and click Next.

Figure 4-1 Properties Dialog for OVF Deployment



> **Note:**
>
> Do not enter plaintext values in the password fields.

11. On the Ready to Complete screen, review the selections, and do one of the following:

- Edit a previous screen by way of the Back button.
- When the edits are completed, or if no edits are required, click Finish

The system displays the Deploying screen.

Figure 4-2 OVF Deployment Status Dialog

When the Deploying screen closes, VM creation and deployment is completed.

5

Configuring Your Device for NFV

Operating the Oracle Communications Session Border Controller as a VNF introduces configuration requirements that define resource utilization by the virtual machine. The applicable configuration elements allow the user to optimize resource utilization based on the application's needs and VM resource sharing.

Oracle Communications Session Border Controller configuration for VNF includes settings to address:

media-manager — Set media manager configuration elements to constrain bandwidth utilization, based on traffic type.

system-config > [*core configuration parameters*] — Set these parameters to specify CPU resources available to DoS, forwarding and transcoding processes.

> **Note:**
>
> Early versions of this software used the bootparameter named **other**, set to **isolcpus=[value]**. Remove this setting for this and all ensuing versions of Oracle Communications Session Border Controller software.

Media Manager Configuration

The Oracle Communications Session Border Controller provides the user with a means of tuning the media manager.

As the Oracle Communications Session Border Controller can classify traffic for use in DoS policing, bandwidth may be reserved for certain traffic types. Reserved bandwidth is expressed as a percentage of maximum available system bandwidth. The system's maximum bandwidth is determined by the hardware configuration and the number of available signaling cores. The maximum system bandwidth is defined as the speed of ingress traffic sent to the host, measured in packets per second (PPS). It is reported in the **show platform limits** command, referring to the "Maximum Signaling rate".

The following configuration options are available in the **media-manager-config**. These options are used to configure reserved bandwidth for application signaling, and ARP, and untrusted traffic.

- **max-signaling-packets**—Set the maximum overall bandwidth available for the host path in packets per second, which includes signaling messages from trusted and untrusted sources. It also includes any Telnet and FTP traffic on media ports. The maximum value for each platform (shown below) is used as the default value for that platform. The valid range is:
  - Minimum-71000
  - Maximum-Depends on platform as shown below
    * For COTs and VM platforms, the maximum is system dependent

- **min-untrusted-signaling**—The minimum percentage of maximum system bandwidth available for untrusted traffic. The rest of the bandwidth is available for trusted resources, but can also be used for untrusted sources per max-untrusted-signaling. Default: 30. Range: 1-100.
- **max-untrusted-signaling**—The percentage of the maximum signaling packets you want to make available for messages coming from untrusted sources. This is a floating highwater mark and is only available when not in use by trusted sources. Default: 100. Range:1-100.
- **tolerance-window**—The size of the window, in seconds, used to measure host access limits for measuring the invalid message rate and maximum message rate for the realm configuration. Default: 30. Range: 0-4294967295.
- **max-arp-rate**—The maximum percentage or max-signaling-packets available for ARP traffic. Default: 30. Range: 1-100.

The user can also set controls on untrusted traffic from either realm or static ACL configuration using the **untrusted-signal-threshold** parameter.

Core Configuration

The user configures CPU core settings using **system-config** parameters. This configuration is based on the specific needs of every individual implementation. These parameters allow the user to set the number of cores they want to assign to forwarding, DoS, and transcoding functionality. The system determines which cores perform those functions, automatically assigning cores for optimal operation in each environment.

The user determines and manages their core configuration based on the services they need. The system allocates cores to signaling upon installation. The user adds forwarding cores to match their needs for handling media. The user also adds DoS and/or transcoding cores if they need those functions in their deployment. The user must then reboot the system for core configuration changes to take effect.

Note the following:

- By default, core 0 is always set to signaling.
- The system selects cores based on function; users do not assign cores.
- The system sets unassigned cores to signaling, with a maximum of 24.

When the user makes core assignments, the Oracle Communications Session Border Controller provides an error message if the system detects an issue. In addition, the system performs a check when the user issues the **verify-config** command to ensure that the total number of forwarding, plus DOS, plus transcoding cores does not exceed the maximum number of physical cores. After the user saves and activates a configuration that includes a change to the core configuration, the system displays a prompt to remind the user that a reboot is required for the changes to take place.

The user can verify core configuration from the ACLI, using the **show datapath-config** command or after core configuration changes during the save and activation processes. When using hyperthreading, which divides cores into a single physical (primary) and a single logical (secondary) core, this display may differ. Bear in mind that the Oracle Communications Session Border Controller binds functions to primary or secondary cores using its own criteria, with secondary cores performing signaling functions only. Hypervisors that provide a view into the type of core assigned to a function allow **show datapath-config** to display primary cores in upper-case letters and secondary cores in lower-case letters. Other hypervisors show all cores as physical.

The Oracle Communications Session Border Controller uses the following lettering (upper- and lower-case) in the ACLI to show core assignments:

- S - Signaling
- D - DoS
- F - Forwarding
- X - Transcoding

The **system-config** element includes the following parameters for core assignment:

- **dos-cores**— Sets the number of cores the system must allocate for DOS functionality. A maximum of one core is allowed.
- **forwarding-cores**—Sets the number of cores the system must allocate for the forwarding engine. Valid values are from 1 to 128.
- **transcoding-cores**—Sets the number of cores the system must allocate for transcoding. Valid values are from 0 to 128, with a default of 0.

To change core assignments, access the **system-config**, as follows.

```
ORACLE# configure terminal
ORACLE(configure)# system
ORACLE(system)# system-config
ORACLE(system-config)#
```

Change existing core assignment settings using the **system-config** parameters listed above. For example, to reserve a core for DoS processing:

```
ORACLE#(system-config) dos-cores 1
```

The Oracle Communications Session Border Controller VNF has no system-based maximum number of cores, other than the range of the **system-config** parameters.

> **Note:**
>
> Refer to the *New in this Release* section at the beginning of this Essentials Guide for any release-specific core configuration constraints.

The system checks CPU core resources before every boot, as configuration can affect resource requirements. Examples of such resource requirement variations include:

- There is at least 1 CPU assigned to signaling (by the system).
- If DoS is required, then there are at least 1 CPU assigned to forwarding and 1 to DoS.
- If DoS is not required, then there is at least 1 CPU assigned to forwarding.

The system performs resource utilization checks every time it boots for CPU, memory and hard-disk to avoid configuration/resource conflicts.

Core configuration is supported by HA. For HA systems, resource utilization on the backup must be the same as the primary.

> **Note:**
>
> The hypervisor always reports the datapath CPU usage fully utilized. This isolates a physical CPU to this work load. However, this may cause the hypervisor to generate a persistent alarm indicating that the VM is using an excessive amount of CPU, possibly triggering throttling. The user should configure their hypervisor monitoring appropriately, to avoid such throttling.

Virtual MAC Addresses for VNFs

Virtual Network Functions (VNFs) rely on their hypervisor environment for MAC address establishment, advertisement and resolution. As such, you cannot derive these addresses using the same method as you do for Acme platforms. For VNFs, Oracle recommends establishing private MAC addressing for virtual MAC address configuration.

To support HA, you configure virtual Ethernet (MAC) address MAC addresses based on the Burned In Addresses (BIA) of the media interfaces. To determine what the virtual MAC addresses should be, you first identify a BIA and then calculate the virtual MACs based on that.

To define the virtual addresses you need to configure for each interface:

1. Identify the base MAC of eth0/wancom0 physical interface using the show interfaces command. For example, in the following display, you can see the base MAC is 00:50:56:C0:00:08:

   ```
   eth(unit number 0):
   Flags: (0x78843) UP BROADCAST MULTICAST ARP RUNNING INET_UP
   Type: ETHERNET_CSMACD
   inet: 111.22.0.123
   Broadcast address: 111.22.255.255
   Netmask 0xffff0000 Subnetmask 0xffff0000
   Ethernet address is 00:50:56:C0:00:08
   ```

2. Set the bottom nibble of the first byte to 2 to define the address as locally administered.
3. Set the top nibble of the first byte to 0 and increment it for each interface.

For example, using the base-MAC for eth0, 00:50:56:C0:00:08, you assign the virtual addresses as follows:

- First media interface virtual MAC = 02:50:56:C0:00:08
- Second media interface virtual MAC = 12:50:56:C0:00:08
- Third media interface virtual MAC = 22:50:56:C0:00:08
- Forth media interface virtual MAC = 32:50:56:C0:00:08

System Shutdown

Use the system's **halt** command to gracefully shutdown the VNF.

```
ACMEPACKET# halt

-----------------------------------------------------
WARNING: you are about to halt this SD!
-----------------------------------------------------
```

```
Halt this SD [y/n]?:
```

See the *ACLI Reference Guide* for further information about this command.

6
Boot Management

Boot Management includes the tasks needed to ensure the system is operating according to the users requirements as it starts up. Requirements met by properly managing system boot include defining management access IP, specifying the load to boot and specifying a system name. The user may set this information manually or configure the operational environment to provide it.

Boot management consists of tasks working with the following:

- Boot Loaders—The user needs to perform file management tasks to ensure that the software used to boot the system is compatible with the application system software itself. This typically includes verifying boot loader and application system software version for compatibility and placing the correct boot loader software in the correct location.
- Boot Parameters—The user sets boot parameters to specify their requirements for boot, including defining management access IP, specifying the load to boot and specifying a system name.
- Boot Flags—The user can, optionally, set special boot parameters called boot flags to further define how the system boots. The user may also set boot flags for diagnostic purposes under the guidance of Oracle support personnel.

Boot Parameters

Boot parameters specify the information that your device uses at boot time when it prepares to run applications.

This section explains how to view, edit, and implement device's boot parameters, and boot flags. Boot parameters:

- Allow you to set the IP address for the management interface (wancom0).
- Allow you to set a system prompt. The target name parameter also specifies the title name displayed in your web browser and SNMP device name parameters.
- Specify the software image to boot and from where the system boots that image.

Boot flags are arguments to a specific boot parameter, and allow functional settings, such as the use of DHCP for acquiring a management port address, as well as various diagnostic startup configurations.

Configuring boot parameters has repercussions on your system's physical and network interface configurations. When you configure these interfaces, you can set values that might override the boot parameters.

The bootparam configuration list is shown below.

```
[Acme Boot]: p
Boot File         : /boot/bzImage
IP Address        : 172.44.12.89
VLAN              :
Netmask           : 255.255.0.0
Gateway           : 172.44.0.1
IPv6 Address      :
```

```
IPv6 Gateway     :
Host IP          :
FTP username     :
FTP password     :
Flags            : 0x00000040
Target Name      : ORACLE
Console Device   : COM1
Console Baudrate : 115200
Other            :


[Acme Boot]: ?
 ?                    - print this list
 @                    - boot (load and go)
 p                    - print boot params
 c                    - change boot params
 v                    - print boot logo with version
 r                    - reboot
 s                    - show license information
```

Boot Parameter Definitions

The system displays all boot parameters when the user configures them after a boot interrupt. The system hides some boot parameters from the ACLI because the user should not configure them. If changed improperly, these parameters can cause the system to fail.

The following table defines each of the parameters that are visible when the user configures after a boot interrupt.

Boot Parameter	Description
Boot File	The name and path of the software image you are booting. Include the absolute path for a local boot from the local /boot volume and for a net boot when a path on the FTP server is needed.
IP Address	IP address of wancom0.
VLAN	VLAN of management network over which this address is accessed. Note: VLANs over management interfaces are not supported on the Acme Packet 6000
Netmask	Netmask portion of the wancom0 IP Address.
Gateway	Network gateway that this wancom0 interface uses.
IPv6 address	Version 6 IP address of wancom0.
IPv6 Gateway	Network gateway that this wancom0 interface uses.
Host IP	IP Address of FTP server from which to download and execute a software image.
FTP Username	FTP server username
FTP password	FTP server password
Flags	Codes that signal the system from where to boot. Also signals the system about which file to use in the booting process. This sequence always starts with 0x (these flags are hexadecimal).
Target Name	Name of the Oracle Communications Session Border Controller as it appears in the system prompt. For example, ORACLE> or ORACLE#. You need to know the target name if you are setting up an HA node. This name is required to be unique among Oracle Communications Session Border Controllers in your network. This name can be 63 characters or less.

Boot Parameter	Description
Console Device	Serial output device type, dependent on platform. COM1 applies to virtual serial consoles, VGA to virtual video console. VGA is the default on VMware and KVM. COM1 is the default on OVM .
Console Baud Rate	The speed in bits per second which the console port operates at. It operates at 115200 BPS, 8 data bits, no stop bit, parity NONE.
Other	Allows miscellaneous and deployment-specific boot settings.

Changing Boot Parameters

You can access and edit boot parameters by using either the ACLI or by interrupting the system boot process.

> **Note:**
>
> Changes to boot parameters do not go into effect until you reboot the system.

Change Boot Parameters from the ACLI

To access and change boot parameters from the ACLI:

1. In Superuser mode, type configure terminal, and press Enter.

   ```
   ORACLE# configure terminal
   ```

2. Type bootparam, and press Enter. The boot device parameters display.

   ```
   ORACLE(configure)# bootparam
   '.' = clear field;  '-' = go to previous field;  q = quit
   Boot File    : /boot/nnScz100.bz
   ```

   To navigate through the boot parameters, press Enter and the next parameter appears on the following line.

   You can navigate through the entire list this way. To go back to a previous line, type a hyphen (-) and press Enter. Any value that you enter entirely overwrites the existing value and does not append to it.

3. To change a boot parameter, type the new value that you want to use next to the old value. For example, if you want to change the image you are using, type the new filename next to the old one. You can clear the contents of a parameter by typing a period and then pressing Enter.

   ```
   ORACLE(configure)# bootparam
   '.' = clear field;  '-' = go to previous field;  q = quit

   Boot File    : /boot/nnSCz100.bz /boot/nnSCz200.bz
   ```

   When you have scrolled through all of the boot parameters, the system prompt for the configure terminal branch displays.

   ```
   NOTE: These changed parameters will not go into effect until reboot.
   Also, be aware that some boot parameters may also be changed through
   PHY and Network Interface Configurations.
   ```

```
ORACLE(configure)#
```

4. Exit the configure terminal branch.
5. Reboot the system for the changes to take effect.

   The ACLI **reboot** and **reboot force** commands initiate a reboot. With the **reboot** command, you must confirm that you want to reboot. With the **reboot force** command, you do not have make this confirmation.

   ```
   ORACLE# reboot force
   ```

   The system completes the full booting sequence. If necessary, you can stop the auto-boot at countdown to fix any boot parameters.

   If you configured boot parameters correctly, the system prompt displays and you can go ahead with configuration, management, or monitoring tasks.

> **Note:**
>
> If you configured the boot parameters incorrectly, the system goes into a booting loop and displays an error message. Press the space bar to stop the loop. Correct the error in the boot parameter, and reboot the system.

Change Boot Parameters by Interrupting a Boot in Progress

To access and change boot parameters by interrupting a boot in progress:

1. When the system is in the process of booting, you can press the space bar on your keyboard to interrupt when you see the following message appear:

   ```
   Press the space bar to stop auto-boot...
   ```

2. After you stop the booting process, you can enter the letter p to display the current parameters, the letter c to change the boot parameters or the @ (at-sign) to continue booting.

   ```
   [Boot]: c
   '.' = clear field;  '-' = go to previous field;  q = quit
   Boot File    : /boot/nnScz100.bz
   ```

   To navigate through the boot parameters, press Enter and the next parameter appears on the following line.

   You can navigate through the entire list this way. To go back to a previous line, type a hyphen (-) and press Enter. Any value that you enter entirely overwrites the existing value and does not append to it.

3. To change a boot parameter, type the new value that you want to use next to the old value. For example, if you want to change the image you are using, type the new filename next to the old one.

   ```
   '.' = clear field;  '-' = go to previous field;  q = quit

   Boot File    : /boot/nnSCz100.bz /boot/nnSCz200.bz
   ```

4. After you have scrolled through the complete list of boot parameters, you return to the boot prompt. To reboot with your changes taking effect, type @ (the at-sign), and press Enter.

```
[Acme Packet Boot]: @
```

The system completes the full booting sequence, unless there is an error in the boot parameters.

If you have configured boot parameters correctly, the system prompt displays and you can go ahead with configuration, management, or monitoring tasks.

> **Note:**
>
> If you have configured the boot parameters incorrectly, the system goes into a booting loop and displays an error message. Press the space bar to stop the loop. Correct the error, and reboot your system.

7

Formatting Disks for VM Deployments

The default SBC VNF template defines a 40GB virtual disk image. The disk is split into System and User areas.

The System area is 20GB, pre-formatted as:

- 2GB **/boot**
- 2GB **/code**
- 16GB **/opt**

This is sufficient for a VM with 8GB RAM. If larger RAM is deployed, the user must reformat the disk with **format system-disk**, setting **/opt** to 2xRAM.

Any additional space is available for the User area. On the default 40GB template, this is 20GB. By default, the User area is not formatted. To make this area available for user data, such as CDRs, the user executes the **format data-disk** command.

> **Note:**
>
> The user can format the System and User areas simultaneously using the **format hard-disk** command.

After drive formatting is complete, the system mounts the newly created partitions.

> **Note:**
>
> If you find that, after the first reboot, the system has not created new partitions, perform another reboot to resolve this issue, which would have been caused by an incorrect dynamic partition table refresh.

Formatting Procedure for VM Deployments

The **format** command requires one of the following arguments:

- system-disk — formats and creates the 2 system partitions: `/opt` and `/opt/crash`
- data-disk — formats and creates 1 or more data partitions with the default (`/mnt/sys` and `/mnt/app`) or user-defined volumes
- hard-disk — formats and creates both the system partition and data partition

After the drive(s) are formatted, the system mounts the newly created partitions.

The following example shows the **format data-disk** command process. In this case, the user formats and mounts the **/sys** and **/app** partitions.

> **Note:**
>
> The format command may only be executed if certain tasks like local CDR and HDR generation are not active. Remove any boot time configuration for these features and reboot the system before attempting to format the hard-disk. In addition, ensure that your device is not passing traffic while you format the any partition.

```
ORACLE# format data-disk

WARNING: Please ensure device is not currently in use by any applications before
proceeding

Continue [y/n]?: y

Use factory default data partitions [y/n]?: y

The following data partitions will now be created:
/sys    3894968320 bytes

/app    35054714880 bytes

Create the data partitions and filesystems as configured above [y/n]?: y

******************************************************
WARNING: All non-system data on the disk will be
permanently erased and unrecoverable.

Are you sure [y/n]?: y

The format process will take a few minutes. Once
the format process begins, it cannot be stopped.
Please do not power down or reboot the system until
the format process is complete.

Continue [y/n]?: y

*** Beginning format process ***

*** Removing previous data partitions - please wait ***

*** Creating new data partitions - please wait ***

*** Formatting partition /sys. Please wait... ***

*** Formatting completed successfully ***

*** Formatting partition /app. Please wait... ***

*** Formatting completed successfully ***

*** Format finished successfully
New partitions have been created ***

*** Mounting partitions ***/sys mounted/app mounted
```

This section of the format hard-drive walk-through shows the data partition creation. The following system output shows that the user has chosen to define a custom data partition scheme by typing "n" at the following prompt.

```
Use factory default data partitions [y/n]?:n
```

In this case, the user creates four partitions.

```
Enter the number of data partitions to create: 4

Total unallocated space = 100 %
Enter the name of volume 1 (or 'q' to quit): partition1
Enter the size of the volume (in %): 20

Total unallocated space = 80 %
Enter the name of volume 2 (or 'q' to quit): partition2
Enter the size of the volume (in %): 40

Total unallocated space = 40 %
Enter the name of volume 3 (or 'q' to quit): partition3
Enter the size of the volume (in %): 20

Total unallocated space = 20 %
Enter the name of volume 4 (or 'q' to quit): partition4
Enter the size of the volume (in %): 20

The following data partitions will now be created:
/partition1    4589934489 bytes

/partition2   9179868979 bytes

/partition3   4589934489 bytes

/partition4    4589934489 bytes

Create the data partitions and filesystems as configured above [y/n]?: y

*****************************************************
WARNING: All non-system data on the disk will be
permanently erased and unrecoverable.

Are you sure [y/n]?: y

The format process will take a few minutes. Once
the format process begins, it cannot be stopped.
Please do not power down or reboot the system until
the format process is complete.

Continue [y/n]?: y

*** Beginning format process ***


*** Removing previous data partitions - please wait ***


*** Creating new data partitions - please wait ***

*** Formatting partition /partition1. Please wait... ***

*** Formatting completed successfully ***
```

```
*** Formatting partition /partition2. Please wait... ***

*** Formatting completed successfully ***

*** Formatting partition /partition3. Please wait... ***

*** Formatting completed successfully ***

*** Formatting partition /partition4. Please wait... ***

*** Formatting completed successfully ***


*** Format finished successfully
New partitions have been created ***


*** Mounting partitions ***

/partition1 mounted
/partition2 mounted
/partition3 mounted
/partition4 mounted
```

Power cycle the system after format is complete. You can re-enable any tasks that may have conflicted with the format, including local CDR and HDR generation, after the power cycle is complete.

8

VM Interfaces

Virtual machines need to interface with their virtual machine management systems to reach and share physical interfaces with other virtual machines. The Oracle Communications Session Border Controller assigns a unique pseudo MAC address from a virtual machine management system and maps it to the Oracle Communications Session Border Controller configuration naming syntax to direct network traffic to physical interfaces based on the type of service they support.

The MACTAB File

Interface mapping on the Oracle Communications Session Border Controller is statically defined by a configuration file named MACTAB.

Sample default Oracle Communications Session Border Controller interface mapping is presented below, via the **show interfaces** command:

```
Device2# show interfaces mapping
Interface Mapping Info----------------------------------------
Eth-IF  MAC-Addr               Label
wancom0 00:50:88:BC:11:12      #generic
wancom1 00:50:88:BC:61:6C      #generic
wancom2 00:50:88:BC:11:C7      #generic
spare   00:50:88:BC:61:12      #generic
s0p0    00:50:88:BC:71:79      #generic
s1p0    00:50:88:BC:21:FF      #generic
s0p1    00:50:88:BC:41:A2      #generic
s1p1    00:50:88:BC:31:AC      #generic
s0p2    FF:FF:FF:FF:FF:FF      #dummy
s1p2    FF:FF:FF:FF:FF:FF      #dummy
s0p3    FF:FF:FF:FF:FF:FF      #dummy
s1p3    FF:FF:FF:FF:FF:FF      #dummy
```

The user creates physical-interface configurations using the names shown under the Eth_IF column, within service configuration elements. There are two types of physical interfaces that apply to the Oracle Communications Session Border Controller, segregated by the naming conventions:

- Media interfaces, shown above as s0p0, s0p1, s1p0 and s1p1
- Management interfaces, shown above as wancom0, wancom1 and wancom2

It is recommended that the user configure physical interfaces using the same naming conventions used in Oracle Session Border Controller that operates on proprietary platforms. These conventions, which simply use 's' for slot and 'p' for port, are visible in the MACTAB file.

The default interface mapping assignment assumes four interfaces on the VM. If deployed with less than four, the user may need to re-assign the resulting interface mapping, which they can verify using the **show interfaces mapping** command after system start-up. If the mapping is wrong, the **interface-mapping>swap** command allows the user to correct it. The most likely change would be to swap the wancom1 mapping with a viable media interface.

Working with the MACTAB File

Interface identification on the Oracle Communications Session Border Controller is based on a system-level file called MACTAB that maps interface MAC addresses to interface naming that can be applied within Oracle Communications Session Border Controller configuration. In most cases, users retain the default mapping. The **show interfaces mapping** command provide access to commands that allow the user see, change and even locate each interface based on this mapping. The MACTAB file is stored in the `/boot` folder. The MACTAB file ensures that interface mapping is persistent, and therefore usable, by your configuration regardless of changes to the system.

The **show interfaces mapping** command displays the current mapping. An example of a MACTAB file that a user has configured is provided below.

```
CMS1# show interfaces mapping
Interface Mapping Info
============================================================
Eth-IF       MAC-Addr               Label
wancom0      00:16:3E:30:00:2A    # ctrl port, onboard MAC
wancom1      00:16:3E:30:00:2B    # 2nd ctrl port, onboard MAC
s0p0         00:16:3E:30:00:2C    # First media interface
s1p0         00:16:3E:30:00:2D    # Second media interrface
============================================================
```

Serial Interfaces

In lieu of IP management access, serial access provides the user with direct access to the Oracle Communications Session Border Controller ACLI. The user must identify how their system allows for serial access. The serial interface can be a critical component of VM interface configuration as the user can make MACTAB changes via the serial interface without interrupting their own access during that management procedure.

Access to the Oracle Communications Session Border Controller serial interface is dependent on platform. Familiarity with the platform is required to understand serial configuration.

Virtual machine management software provides a simulated serial port service from which the user gets direct serial access to each system. See your virtual machine manager's documentation for instructions on accessing these serial ports.

Serial port configuration, via boot parameters, is the same across all platforms.

9

References and Debugging

Packet Trace Over Systems using DPDK

The Oracle Communications Session Border Controller's packet trace tool provides the user with the ability to capture traffic from the Oracle Communications Session Border Controller itself. The packet capture command is documented elsewhere, but the syntax and operation for systems using DPDK is not the same.

There are two capture modes across the product line, one that saves traffic locally and one that mirrors traffic to a user-specified target. Software only deployments, including S-CZ7.3.9, support local capture only.

The user invokes the tool from the ACLI, manually specifying:

- How to capture (local only for S-CZ7.3.9)
- What to capture
- Capture start and stop

Local capture supports PCAP filters to specify the signaling traffic you want to capture. The default packet trace filter uses the specified interface to capture both ingress and egress traffic. The user can then use the **packet-trace** command's syntax to filter based on target IP ad well as local and remote port. To further specify captured traffic, the user can also append the command with a PCAP filter enclosed in quotes. PCAP filter syntax is widely published.

The user can run only a single capture on a given interface. However, the user can run multiple captures simultaneously, as long as they are on separate interfaces.

Local packet capture is dependent on access control configuration, not capturing any denied traffic.

> **Note:**
>
> Do not run packet-trace simultaneously with other Oracle Communications Session Border Controller replication features, such as LI, SRS, SIP Monitoring and Trace, and Call Recording. These features may interfere with each other, corrupting each's results.

Packet Trace Local enables the Oracle Communications Session Border Controller to capture traffic between two endpoints, or between itself and a specific endpoint. To accomplish this, the Oracle Communications Session Border Controller replicates the packets sent and received and saves them to disk in .PCAP format.

By default, the system saves the .PCAP file in /opt/traces, naming it with the applicable interface name as well as the date and time of the capture. Alternatively, the user can specify file name using the system supports the PCAP filter flags -w.

The system rotates the PCAPs created in this directory by size. The last 25 files are kept and are rotated when they reach 100 MB. If there are capture files in the /opt/traces directory when

this command is run, the system prompts the user to remove them before running new captures. If preferred, the user can decline this file deletion.

Starting a Local Packet Trace on Systems Running DPDK

You use the start a packet trace by entering the appropriate ACLI command with these pieces of information:

- Network interface (name:subport ID combination)
- (Optional) IP address to be traced; if you do not enter local and/or remote ports when you start the trace, the Oracle Communications Session Border Controller traces all open sockets.
- (Optional) Local UDP/TCP port on which the Oracle Communications Session Border Controller sends and receives traffic to be traced.
- (Optional) Remote UDP/TCP port to which the Oracle Communications Session Border Controller sends traffic, and from which it receives traffic to be traced; you cannot enter the remote port without specifying a local port.
- (Optional) Enter a tcpdump command line within quotes.

Note that the system supports local packet trace on all platforms. To start a packet trace with local and remote ports specified:

1. Enter the ACLI **packet-trace local** command followed by a Space, and the parameter **start**. After another space, type in the name and subport ID for the network interface followed by a Space.

   The syntax below includes the IP address to be traced, the local port number, then the remote port number separated by spaces.

2. Press Enter.

```
ORACLE# packet-trace local start core:0 192.168.10.99 5060 5060
Trace started for 192.168.10.99
```

Stopping a Local Packet Trace on Systems Running DPDK

You stop a local packet trace by entering the appropriate ACLI command with these pieces of information:

- Network interface (name:subport ID combination)
- (Optional) IP address to be traced
- (Optional) Local UDP/TCP port on which the Oracle Communications Session Border Controller sends and receives traffic to be traced
- (Optional) Remote UDP/TCP port to which the Oracle Communications Session Border Controller sends traffic, and from which it receives traffic to be traced

If the packet trace you want to stop has no entries for local and/or remote ports, then you do not have to specify them.

1. To stop a packet trace with local and remote ports specified, enter the ACLI **packet-trace local** command followed by a Space, and the word **stop**. After another Space, type in the name and subport ID for the network interface followed by a Space, the IP address to be traced followed by a Space, the local port number followed by a Space, and then optionally the remote port number. Then press Enter.

```
ORACLE# packet-trace local stop core:0 192.168.10.99 5060 5060
```

2. To stop all packet traces on the Oracle Communications Session Border Controller, enter the ACLI **packet-trace local** command followed by a Space, and the word **stop**. After another Space, type the word **all** and press Enter.

```
ORACLE# packet-trace local stop all
```

SNMP MIBs and Traps

The following MIBs and traps are supported for the Oracle Communications Session Border Controller. Please consult the Oracle Communications Session Border Controller MIB Reference Guide for more SNMP information.

apUsbcSysDPDKObjects

This group of objects, found in the `ap-usbcsys.mib`, provide a listing of DPDK statistics.

MIB Object	Object ID: 1.3.6.1.4.1.9148.3.17.1.1.13 +	Description
apUsbcSysDPDKFwdPurpose	.1	A bitset representing Forwarding cores. 1s represent forwarding cores, while 0s represent non-forwarding cores.
apUsbcSysDPDKDOSPurpose	.2	A bitset representing DoS cores. Bits set to 1 represent DoS cores, while 0s represent non-DoS cores.
apUsbcSysDPDKSigPurpose	.3	A bitset representing signaling cores. Bits set to 1 represent signaling cores, while 0s represent non-signaling cores.
apUsbcSysDPDKTransPurpose	.4	A bitset representing transcoding Cores. Bits set to 1 represent transcoding cores, while 0s represent non-transcoding cores.
apUsbcSysDPDKCmdLine	.5	System CmdLine string - as defined in /proc/cmdline. (including relevant bootparams.)
apUsbcSysDPDKFileMem	.6	Total DPDK File Memory.
apUsbcSysDPDKSysMem	.7	Total DPDK System Memory
apUsbcSysDPDKNum1G	.8	Number of 1GB Hugepages allocated.
apUsbcSysDPDKNum2MB	.9	Number of 2MB hugepages allocated.
apUsbcSysDPDKHypervisorType	.10	The description regarding the system type and what hypervisor the system is running on (OVM, KVM, VMWare,...).
apUsbcSysDPDKAddFwdCores	.11	Number of additional cores that may be used for forwarding.
apUsbcSysDPDKAddSigCores	.12	Number of additional cores that may be used for signaling.

MIB Object	Object ID: 1.3.6.1.4.1.9148.3.17.1.1.13 +	Description
apUsbcSysDPDKAddTransCores	.13	Number of additional cores that may be used for transcoding.

apUsbcSysScalingObjects

This group of objects, found in the `ap-usbcsys.mib`, provide a listing of objects relating to scaling VMs.

MIB Object	Object ID: 1.3.6.1.4.1.9148.3.17.1.1.12+	Description
apUsbcSysEstSessions	.1	Estimated number of unecrypted media sessions.
apUsbcSysEstG711G729Trans	.2	Estimated number of G711<->G729 transcoded media sessions.
apUsbcSysEstSigTPS	.3	Estimated number of signaling TPS.
apUsbcSysEstACLs	.4	Estimated number of ACLs.
apUsbcSysEstTCP	.5	Estimated number of TCP connections.
apUsbcSysEstTLS	.6	Estimated number of TLS connections.
apUsbcSysEstVLANs	.7	Estimated number of VLANs.

Show Commands

show datapath-config

This command displays the current DPDK-based settings for the datapath. The command has no arguments.

Syntax

```
show datapath-config
```

Output Example

As shown below, this command displays key core and memory configuration of the deployment. Exact output is dependent on your environment.

For example, you may or may not be using hyperthreading, which establishes physical vs. logical CPU considerations. Some hypervisors do not present physical and logical cores such that the SBC can differentiate between CPU types using uppercase and lowercase letters. In these conditions, the SBC displays all CPUs with uppercase letters. When supported by the platform, however, the SBC displays a second logical core of each physical core using lowercase lettering.

Bear in mind that the second, logical core of a physical core cannot be assigned to any DPDK function. The system automatically sets them to signaling.

A VM with 8 vCPUs and hyper-threading enabled assigns the pairing of the cores as follows:

- Cores (0,4)
- Cores (1,5)
- Cores (2,6)
- Cores (3,7)

```
NFV-1# show datapath-config
 Number of cores assigned to VM: 8
        Current core assignment: S-F-D-X-s-s-s-s
            Requested Page size: 1 GB
              Current Page size: 2 MB
           Number of 1 GB pages: 0
           Number of 2 MB pages: 10
            Total system memory: 3841 MB
   Memory reserved for datapath: 1020 MB
  Memory requested for datapath: 2048 MB
```

Field	Description
Number of cores assigned to VM	The number of cores that this VM can use.
Current core assignment	The core assignment, based on function type: • S - signaling • D - DoS • F - Forwarding • X - Transcoding The position of the function indicates the core number.
Requested Page size	Configured page size.
Current Page size	Actual page size.
Number of 1 GB pages	Number of 1GB Hugepages allocated.
Number of 2 MB pages	Number of 1GB Hugepages allocated.
Total system memory	Total DPDK System Memory.
Memory reserved for datapath	Actual memory reserved for datapath.
Memory requested for datapath	Configured memory reserved for datapath.

show platform limits

This command displays the current limits for a variety of operating capacities. The output of **show platform limits** is based on the platform this command is executed from and the software version running. The command has no arguments.

Syntax

Sample output is displayed below.

```
ORACLE# show platform limits
Maximum number of sessions:3000
Maximum number of ACLS: 60000
```

```
Maximum number of common PAC buffers: 8000
Maximum number of kernel-rules: 216256
Maximum CPS rate: 300
Maximum number of TCP Connections: 60000
Maximum number of TLS Connections: 10
Maximum number of packet buffers: 30000
Maximum Signaling rate: 4000
Maximum number of session agents: 125
Maximum number of System ACLs: 256
Maximum number of VLANs: 4096
Maximum number of ARPs: 4104
Maximum number of INTFC Flows: 4096
Maximum number of Static Trusted Entries: 8192
Maximum number of Untrusted Entries: 4096
Maximum number of Media Entries: 6000
Maximum number of Deny Entries: 8192
Maximum number of Internal Flows: 32
Maximum number of Sip Rec Sessions: 512
Maximum number of RFC 2833 Flows: 6000
Maximum number of SRTP Sessions: 500
Maximum number of QoS Sessions: 3000
Maximum number of Xcoded Sessions: 100
Maximum number of HMU Flows: 6000
Maximum number of Transport Sessions: 0
Maximum number of MSRP Sessions: 0
Maximum number of SLB Tunnels: 0
Maximum number of SLB Endpoints: 0
Maximum number of IPSec SAs: 0
Maximum Licensed Capacity: 256000
```

Supporting Configuration

The following configuration elements which are not mentioned in this guide are required for the Oracle Communications Session Border Controller to function. Please refer to the Oracle Communications Session Border Controller *ACLI Configuration Guide* for details about configuring all supporting elements.

- network-interface
- physical-interface
- sip-interface
- realm-config
- sip-config
- system-config

The following configuration elements are mentioned in this guide briefly. Refer to the Oracle Communications Session Border Controller *ACLI Configuration Guide* for details about these object's configuration:

- local-policy
- session-agent

Log Files for the VNF

The following log files capture log messaging related to VNF:

- log.usdp - Contains DPDK processing log messages.
- log.usdpClient - Contains log messages related to overall system.

Refer to Oracle Support personnel for details on working with these files and their content.

A

Known Issues and Caveats

Known Issues

This table lists S-Cz7.3.9 Known issues. The user can reference defects by Service Request number and can identify the issue, any workaround, when the issue was found, and when it was fixed using this table. Issue descriptions not carried forward from previous versions' Release Notes and documented herein are not relevant to this release. The user can review delivery information, including defect fixes via this release's Build Notes.

System

ID	Description	Found In	Fixed In
	This release of the system does not support in-service software upgrades. Workaround: If moving to 7.3.9 from a previous version, delete the existing VM from the hypervisor entirely and build the new VM from scratch.	S-Cz7.3.9	
	The Oracle Communications Session Border Controller does not support overlapping IP addresses on Media interfaces between end points and SBCs in deployments that use an Oracle Communications Subscriber Aware Load Balancer for load balancing.	S-Cz7.3.9	
	Media and management (wancom) interfaces may now be configured within the same subnet, regardless of VLAN. All interfaces must share the same subnet mask and default gateway.	S-Cz7.2.9	S-Cz7.3.9
	In some circumstances when running VLANs on IPv6 interfaces, the SBC responds to ping requests out the wrong interface.	S-Cz7.3.9	
	When running the SBC over OVM using SR-IOV interface mode, the user may find that ping requests fail, responding with "Request not Found".	S-Cz7.3.9	
	When running the SBC over OVM Server 3.4.2 (Oracle X5-2), X540 (ixgbe) driver, and VLANs, and collecting a local packet-trace, the SBC replies to ping requests twice.	S-Cz7.3.9	
	In some cases when using the ixgbe driver, the user may find that PING requests and responses fail over media interfaces, despite succeeding over management interfaces.	S-Cz7.3.9	
	The SBC may respond to IPSec requests with unencrypted responses.	S-Cz7.3.9	
	When running the SBC over KVM or Vmware using PCI-Passthrough interface mode, the user may find that SRTP does not work.	S-Cz7.3.9	

ID	Description	Found In	Fixed In
	When running the SBC over KVM using PCI-Passthrough interface mode, the user may find that the linux host becomes unresponsive. Workaround - The user should review their kernel version and, if applicable, downgrade to UEK-4u1 version.	S-Cz7.3.9	
	The system may display Type: UNKNOWN_TYPE for media interfaces within the show interfaces command.	S-Cz7.3.9	

CPU-Core

ID	Description	Found In	Fixed In
	When configured with more than 1 forwarding core, calls on the Oracle Communications Session Border Controller may not be balanced across all forwarding cores. Workaround: Call load is balanced across available forwarding cores using a hashing algorithm. If a significant percentage of calls originate from the same source IP address, hashing may not be able to efficiently balance the load. Customers should engineer their network to utilize multiple endpoints to avoid such scenarios.	S-Cz7.2.9	S-Cz7.3.9

SRTP

ID	Description	Found In	Fixed In
	The system corrupts SRTP packets that it sends out without the authentication tag. Workaround: Use SRTP with authentication.	S-Cz7.3.9	
	When running the SBC over KVM or Vmware using PCI-Passthrough interface mode, the user may find that SRTP does not work.	S-Cz7.3.9	

Caveats

This section lists caveats to be aware of when deploying the Oracle Communications Session Border Controller.

Functional Limitations

The list below addresses broad SBC functionality not available in this release, including:

- Native transcoding for codecs other than G.711 and G.729.
  Workaround: For all other codecs, configure your environment and system for pooled transcoding.
- Inband DTMF detection
- DTMF generation
- RTCP detection
- RTCP generation for G.711 or G.729
- DDoS for IMS-AKA

- MSRP functionality
- TSCF functionality
- H.323 signaling or H.323-SIP inter-working
- In-service software upgrades for KVM and OVM deployments
- The SIP Monitor & Trace and WebGUI features are unsupported. Ensure that the **system> web-server-config> state- parameter** is set to disabled.

General Caveats

This section lists functional caveats applicable to this release.

- This release of the system does not support the following IPSec components:
  - IKEv1
  - Authentication header (AH)
  - The AES-XCBC authentication algorithm
  - Dynamic reconfiguration of security-associations
  - Hitless HA failover of IPSec connections.
- The OVM server 3.4.2 does not support the virtual back-end required for para-virtualized (PV) networking. VIF emulated interfaces are supported but have lower performance. Consider using SR-IOV or PCI-passthru as an alternative if higher performance is required.
- Default levels for scalability and are set to ensure appropriate throttling based on platform capacity factors such as hypervisor type, number and role of CPU cores, available host memory and I/O bandwidth. In some cases, those defaults may not be appropriate and throttling may occur at lower or higher call rates than expected. Please contact Oracle Technical Support for details on how to override the default throttles, if required.
- To support HA failover, MAC anti-spoofing must be disabled for media interfaces on the host hypervisor/vSwitch/SR-IOV_PF.
- For the Netra X5-2, the VNF currently only supports the X710 in the onboard slot for Management and HA and the 82599 in the expansion slot, when using KVM. This is a kernel defect; the user can monitor the kernel errata to uplift this fix when it becomes available.
- Attempting to use more than 6 cores for more than ~26k sessions results in RX buffer leaks. If this traffic rate persists, buffers become exhausted and the system crashes.

Upgrade Caveat

When upgrading from S-CZ7.3.9p3 (or earlier) to S-CZ7.3.9p4 or later, you must also update the target release's companion stage 3 bootloader.

B

DPDK-Based OVS Installation

System Design and/or Deployment Engineering may decide to use Open V-Switch (OVS) within Data Path Development Kit (DPDK) architectures. This section explains how to create a DPDK-based OVS Installation. This installation also sets up multi-queues using **dpdkvhostuser** for media traffic. The procedure below is an example; step details vary based on deployment environment.

The procedure below assumes the virtual SBC (vSBC) is installed and ready to start.

1. Install Fedora 23 from a CD or an ISO image.
2. Change the kernel boot parameters by adding the following arguments to the **/etc/default/grub** file:

   ```
   hugepagesz=1G
   hugepages=64
   default_hugepagesz=1G
   isolcpus=1-2
   ```

3. Update the installed Fedora 23 packages. If your network is behind a proxy, set the value of proxy in **/etc/dnf/dnf.conf**.

   ```
   proxy=http://proxy.example.com
   ```

4. Update your system.

   ```
   dnf -y update
   ```

5. Install additional required packages.

   ```
   dnf groupinstall -y "Development Tools"

   dnf install -y hwloc gcc-c++ openssl-devel glib autoconf libtool
   glib2-devel zlibrary gtk+-devel gtk3-devel tunctl kernel-devel
   ```

6. Set up environment variables.

   ```
   export DPDK_DIR=/usr/src/dpdk
   export DPDK_BUILD=$DPDK_DIR/x86_64-native-linuxapp-gcc/
   export OVS_DIR=/usr/src/ovs
   export DB_SOCK=/usr/local/var/run/openvswitch/db.sock
   export QEMU_DIR=/usr/src/qemu-2.6.0-rc3
   ```

7. Download and install the following in the **/usr/src** directory.
   - dpdk-16.04.tar.gz
   - openvswitch-ovs-v1.1.0pre2-10269-gac93328.tar.gz (April 27th snapshot version)
   - qemu-2.6.0-rc3.tar.bz2

   a. Change directories to `/usr/src`.

      ```
      cd /usr/src
      ```

   b. Install DPDK.

      ```
      git clone http://dpdk.org/git/dpdk
      cd dpdk
      ```

```
echo "CONFIG_RTE_BUILD_COMBINE_LIBS=y" >> config/common_linuxapp
make install T=x86_64-native-linuxapp-gcc DESTDIR=install
cd ..
```

> **Note:**
>
> If `make` fails because it cannot find `/lib/modules/<version>/build`, run `dnf -y install kernel-devel-<version>` and run `make` again.
>
> ```
> dnf -y install kernel-devel-4.5.5-300.fc24.x86_64
> ```

c. Install OVS.

```
git clone https://github.com/openvswitch/ovs
cd ovs
./boot.sh
./configure --with-dpdk=$DPDK_BUILD CFLAGS="-g -O2 -Wno-cast-align"
make install
cd ..
```

> **Note:**
>
> If you get the error message `ImportError: No module named six`, make sure your http_proxy variable is set and run `pip install six --proxy=$http_proxy`.

> **Note:**
>
> If `configure` returns an error, see the INSTALL file for your distribution.

d. Install qemu.

```
wget http://wiki.qemu.org/download/qemu-2.6.0-rc3.tar.bz2
tar jxf qemu-2.6.0-rc3.tar.bz2
cd qemu-2.6.0-rc3
./configure
make install
cd ..
```

8. Install DPDK modules.

```
modprobe uio insmod $DPDK_BUILD/kmod/igb_uio.ko
```

9. Mount the hugepages.

```
mount -t hugetlbfs -o pagesize=1G none /dev/hugepages
```

10. Choose the host NICs for guest VM's media traffic. These interfaces need to be bound to DPDK. For example, you may choose **eno1** as the vSBC's access interfaces, s0p0.

    Do not use host DPDK interfaces as vSBC wancom interfaces.

11. Assign these media traffic NICs to OVS DPDK.

```
$DPDK_DIR/tools/dpdk_nic_bind.py --bind=igb_uio eno1
$DPDK_DIR/tools/dpdk_nic_bind.py --bind=igb_uio eno2
```

12. Create OVS directories.

```
mkdir -p /usr/local/etc/openvswitch
mkdir -p /usr/local/var/run/openvswitch
rm -f /usr/local/etc/openvswitch/conf.d
```

13. Create the OVS database schema.

```
ovsdb-tool create /usr/local/etc/openvswitch/conf.db /usr/local/share/
openvswitch/vswitch.ovsschema
```

14. Start the ovsdb server/ovsvswitch.

```
ovsdb-server --remote=punix:$DB_SOCK --
remote=db:Open_vSwitch,Open_vSwitch,manager_options     --private-
key=db:Open_vSwitch,SSL,private_key     --
certificate=Open_vSwitch,SSL,certificate     --bootstrap-ca-
cert=db:Open_vSwitch,SSL,ca_cert --pidfile --detach

ovs-vsctl --no-wait init

ovs-vswitchd  --dpdk -vhost_sock_dir /usr/local/var/run/openvswitch -c 0xe -
n 4 --socket-mem 16384,0 -- unix:$DB_SOCK --pidfile --detach --log-file=/usr/
local/var/log/openvswitch/ovs-vswitchd.log
```

15. Set up the CPU mask so that these cores are used for pmd threads in the ovsvswitch.

```
ovs-vsctl set Open_vSwitch . other_config:pmd-cpu-mask=6
```

    This starts two PMD threads, and therefore two RX/TX queue pairs.

16. Create bridges and ports. (You can create multiple ports to launch multiple guest VMs.)

```
ovs-vsctl add-br eno1br -- set bridge eno1br datapath_type=netdev
ovs-vsctl add-port eno1br dpdk0 -- set Interface dpdk0 type=dpdk
ovs-vsctl add-port eno1br vhost-user-port0-on-eno1br -- set Interface vhost-
user-port0-on-eno1br type=dpdkvhostuser
ovs-vsctl set Interface vhost-user-port0-on-eno1br options:n_rxq=2

ovs-vsctl add-br eno2br -- set bridge eno2br datapath_type=netdev
ovs-vsctl add-port eno2br dpdk1 -- set Interface dpdk1 type=dpdk
ovs-vsctl add-port eno2br vhost-user-port0-on-eno2br -- set Interface vhost-
user-port0-on-eno2br type=dpdkvhostuser
ovs-vsctl set Interface vhost-user-port0-on-eno2br options:n_rxq=2

ovs-vsctl add-br ens1f2br -- set bridge ens1f2br datapath_type=netdev
ovs-vsctl add-port ens1f2br ens1f2
ip tuntap add mode tap tap0-on-ens1f2br
ip link set dev tap0-on-ens1f2br up
ovs-vsctl add-port ens1f2br tap0-on-ens1f2br
```

17. Export your display.

```
export DISPLAY=IPADDRESS_WHERE_YOU_WANT_TO_SEE_QEMU_GRAPHICAL_OUTPUT:1
```

18. Launch your VM.

```
/usr/local/bin/qemu-system-x86_64 -name vsbc1 -S -machine pc-
i440fx-2.1,accel=kvm,usb=off -cpu SandyBridge -m 8192 -realtime mlock=off \
-smp 4,sockets=1,cores=4 \
-rtc base=utc -no-shutdown -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=0x5 \
-drive file=/usr/src/vms/nnSCZ739a6-img-vm_kvm.qcow2,if=none,id=drive-
ide0-0-0,format=qcow2,cache=none \
```

```
-device ide-hd,bus=ide.0,unit=0,drive=drive-ide0-0-0,id=ide0-0-0,bootindex=1
\
-device virtio-balloon-pci,id=balloon0,bus=pci.0,addr=0x6 \
-monitor stdio \
-netdev tap,script=no,id=mgmt,ifname=tap0-on-ens1f2br \
-device e1000,netdev=mgmt,mac=52:55:00:dd:ee:01 \
-chardev socket,id=char1,path=/usr/local/var/run/openvswitch/vhost-user-
port0-on-eno1br \
-netdev type=vhost-user,id=access,chardev=char1,vhostforce=on,queues=2 \
-device virtio-net-pci,netdev=access,mac=52:55:00:dd:ee:02,mq=on,vectors=6 \
-chardev socket,id=char2,path=/usr/local/var/run/openvswitch/vhost-user-
port0-on-eno2br \
-netdev type=vhost-user,id=core,chardev=char2,vhostforce=on,queues=2 \
-device virtio-net-pci,netdev=core,mac=52:55:00:dd:ee:03,mq=on,vectors=6 \
-object memory-backend-file,id=mem,size=8192M,mem-path=/dev/
hugepages,share=on \
-numa node,memdev=mem -mem-prealloc
```

You see the VM display in a different window. You launch VMs from your qemu control monitor.

19. Execute the **'cont'** command in the qemu control monitor to boot the vSBC in the guest VM.

```
command 'cont'
```