HP Helion OpenStack Carrier Grade 2.1: KVM Region Administration Guide

Contents

Preface

The HP Helion OpenStack Carrier Grade Administration Guide is one of several documents released with the HP Helion OpenStack Carrier Grade product line. They include:

- Quick Start Guide
- Software Installation Guide
- Reference Deployment Scenarios
- System Engineering Guidelines
- Release Notes

The Administration Guide is addressed to HP Helion OpenStack Carrier Grade administrators responsible for the following areas:

- Creating and maintaining resources used by end users (tenants), such as virtual machines, storage pools, network objects, and access credentials.
- Operating and maintaining components of the HP Helion OpenStack Carrier Grade product, such as clusters, nodes, system software, licensing files, and backup and restore facilities.

Familiarity with these technologies is assumed:

- The OpenStack cloud computing service on which HP Helion OpenStack Carrier Grade is built. This guide provides information specific to HP Helion OpenStack Carrier Grade. Detailed discussion of OpenStack components is outside its scope. The OpenStack documentation at http://docs.openstack.org can be used as a complement.
- Networking communication standards and protocols, required to manage the physical and virtual networks deployed within OpenStack and the HP Helion OpenStack Carrier Grade product.
- Basic computing technology, including familiarity with BIOS settings, network interfaces, and drive configurations, required to manage the physical nodes in the cluster.
- The Linux operating system, including its file system, environment, and associated command line operations.

Before installing the software, you are strongly encouraged to read and understand the Overview and Planning chapters in this guide. These chapters provide a mandatory introduction to the concepts and considerations required for a smooth and successful installation.

For the most part, the Administration Guide assumes that the HP Helion OpenStack Carrier Grade software is already installed, and that you can log in to the web administration interface and the node consoles.

Chapter 1

HP Helion OpenStack Carrier Grade Overview

Topics:

HP Helion OpenStack Carrier Grade is a high-performance, high-availability, cloud operating system that enables telecommunications operators to use Commercial-off-the-shelf (COTS) hardware to manage Virtualized Network Functions (VNF) within a carrier grade Network Function Virtualization (NFV) architecture.

HP Helion OpenStack Carrier Grade brings together the flexibility and scalability of the IT cloud, and the high-availability and performance demanded by the Telecommunications industry, into a unique carrier grade, industry-leading solution to deliver a price-performance ratio well above alternative solutions. HP Helion OpenStack Carrier Grade is aligned with the ETSI-NFV architecture.

Core HP Helion OpenStack Carrier Grade Capabilities

HP Helion OpenStack Carrier Grade leverages core KVM virtualization and carrier-grade technologies, Intel DPDK high-performance packet processing, open architectures, and the OpenStack software suite. It extends the OpenStack software suite to implement a unique carrier-grade, high-availability architecture, on which high-performance production systems can be deployed.

Core capabilities unique to HP Helion OpenStack Carrier Grade include:

High-Performance Networking

HP Helion OpenStack Carrier Grade provides significantly improved network performance over the default open source OpenStack solution, up to 20 Gbps throughput through a guest application on a virtual machine.

At the center of HP Helion OpenStack Carrier Grade Networking is a DPDK-Accelerated Virtual L2 Switch (AVS), running on the compute node hosts. It provides connectivity between virtual machines on the same or different compute nodes, and between virtual machines and external networks. AVS supports a variety of network connectivity options with the hosted virtual machines, as follows:

- Unmodified guests can use Linux networking and virtio drivers. This provides a mechanism to bring existing applications into the production environment immediately.
- Guest applications can be modified to leverage the Accelerated Virtual Port (AVP) drivers provided with HP Helion OpenStack Carrier Grade. AVP ports provide increased throughput over plain virtio drivers when connected to the AVS.
- For the highest performance, guest applications can also be modified to make use of the Intel DPDK libraries and the AVP poll-mode

drivers provided by HP Helion OpenStack Carrier Grade to connect with the AVS Switch.

In addition to AVS, HP Helion OpenStack Carrier Grade incorporates DPDK-Accelerated Neutron Virtual Router L3 Forwarding (AVR). Accelerated forwarding is used between directly attached tenant networks and subnets, as well as for gateway, SNAT, and floating IP functionality.

HP Helion OpenStack Carrier Grade also supports direct guest access to NICs using PCI passthrough or SR-IOV, with enhanced NUMA scheduling options compared to standard OpenStack. This offers very high performance, but because access is not managed by HP Helion OpenStack Carrier Grade or the vSwitch process, there is no support for live migration, Titanium Server-provided LAG, host interface monitoring, QoS, or ACL. If VLANs are used, they must be managed by the guests.

High-Availability

HP Helion OpenStack Carrier Grade provides a number of extensions to OpenStack to deliver highly available hosting of virtual machines, including:

1:1 OpenStack controller services

Automatic configuration of the OpenStack controller services in 1:1 active/standby mode across two controller nodes.

Fast detection of compute host failures

Implemented by using a highly efficient and highly scalable heartbeat protocol between the controller and compute nodes.

Fast recovery of virtual machines instances upon detection of a compute node failure

Extensions to Nova services that automatically re-schedule impacted virtual machine instances to available alternative compute nodes in the cluster.

Fast recovery of tenant network services upon detection of a compute node failure

Extensions to Neutron that automatically re-schedule impacted network services such as DHCP, L3 Routing, and User/Meta data server, for all affected tenant networks. This covers tenant networks spanning multiple compute nodes.

Fast and enhanced detection of virtual machines failures

Failure of any KVM/QEMU instance is automatically reported by the Linux death process notification.

Additionally, modified guests can make use of the HP Helion OpenStack Carrier Grade Guest Heartbeat Library to register application-specific health check callbacks with the virtual machine. This registration allows the compute node to monitor the health of the guest application. The frequency of the health checks is determined at registration time. The semantics of the health check, that is, determining when the application is in a good or a bad state, is under the control of the application itself.

Automatic recovery of failed virtual machine instances

Extensions to Nova that automatically re-start a failed virtual machine on the same compute node. If the re-start operation fails, the virtual machine is re-scheduled to use an alternative compute node.

Enhanced virtual machine server groups

HP Helion OpenStack Carrier Grade enhances virtual machine server groups by adding the following attributes:

- maximum number of virtual machine instances
- best effort or strict affinity policy

Virtual machines in the same server group can also use a HP Helion OpenStack Carrier Grade server group messaging API for low-bandwidth communications.

Live migration support with DPDK-accelerated networking

Live migration support for virtual machines, using the high-performance networking options included with HP Helion OpenStack Carrier Grade.

Graceful shutdown (and other operations) of virtual machines

Nova extensions that turn the default shutdown operation of virtual machines into an ACPI shutdown. Guest applications can therefore register shutdown scripts using standard ACPI mechanisms to execute operations such as closing files, updating persistent databases, or cleanly disconnecting from subscribed services.

Optionally, guest applications can use the HP Helion OpenStack Carrier Grade Guest Heartbeat Library to register to receive process management requests such as shutdown, live-migrate, pause, auto-scale, and others. The guest applications can then reject the requests based upon application-specific state directives, or prepare for a graceful execution.

Link Aggregation (LAG) support

Support for LAG (with LACP), also known as Aggregate Ethernet, on controller and compute nodes for link protection.

Protected HA Middleware

HP Helion OpenStack Carrier Grade HA middleware is protected by a process monitor that uses Linux death process notification to detect and respawn critical HA processes.

Virtual Machine Performance

HP Helion OpenStack Carrier Grade provides an improved execution environment for virtual machines that contributes to their overall performance when deployed on the compute nodes. Enhancements to the execution environment include:

- Low-latency, bounded delivery of virtual interrupts, and availability of low-latency high resolution timers. This is done by leveraging the optimized kernel and KVM/QEMU implementations of the Open Virtualization Platform (OVP).
- Option to allocate 2 MB or 1 GB huge memory pages to reduce swapping and TLB lookup misses.

- Improved resource tracking of VMs that use dedicated CPUs, to ensure successful migration and evacuation.
- Support for affining guest NUMA nodes to specific host NUMA nodes.
- Capability to specify specific CPU models to be used by a virtual machine in order to leverage advanced features of the CPU architecture.

Guest VLANs

HP Helion OpenStack Carrier Grade guests can make use of Titanium Server-defined VLANs to encapsulate IP traffic from a single or multiple IP subnets on a virtual Ethernet interface. HP Helion OpenStack Carrier Grade tenants can define one or more VLAN-tagged IP subnets on a single tenant network to allow their guests' traffic to be encapsulated on VLAN IDs of their choice. These TiS-defined VLAN-tagged IP subnets have access to all of the services of the virtualized network infrastructure, such as DHCP, virtual routing, meta-data server, etc.

Alternately, HP Helion OpenStack Carrier Grade guests can make use of transparent VLANs, in which packets are encapsulated within a provider network segment without removing or modifying the guest VLAN tag(s). VLAN Transparent provides more flexibility with respect to how VLAN tagged packets are handled without requiring that you pre-define which VLAN instances will be used. A guest can send/receive VLAN tagged packets (802.1q) and/or double VLAN tagged packets (802.1ad) without first defining a VLAN-tagged subnet in HP Helion OpenStack Carrier Grade. In addition, a VLAN transparent network will also propagate VLAN priority information (802.1p) to the destination VM instance. However, as a consequence of allowing arbitrary VLAN instances, you loose the ability to access DHCP servers, virtual routers, or meta-data servers over those VLAN-tagged networks.

Support for Guest VLANs is a key capability that facilitates the porting of current user applications running on dedicated physical servers using VLANs to a virtualized environment.

Inventory Management

HP Helion OpenStack Carrier Grade provides complete inventory management of hosts in the OpenStack cloud, allowing the system administrator to install, configure, and maintain the individual servers. The inventory service gives the system administrator the following capabilities:

- Host management
  - discovery of new hosts in the cluster
  - installation of the appropriate load, controller, compute, or storage images
  - configuration of the management, OAM, infrastructure, and provider network interfaces on each host
  - creation and use of data interface and CPU profiles to simplify host configuration
  - configuration of the number of CPUs allocated to the Accelerated Virtual Switch (AVS) on each compute node
  - configuration of huge memory page allocations for VM use
  - hardware sensors (for example, temperature or voltage sensors)
- Administrative operations

  - lock/unlock
  - switch active controller host or service
  - reboot and reset host
  - power-on and power-off
  - software re-installation
- Status reporting
  - admin state
  - operational state
  - availability state
  - uptime
  - real-time command execution reports, such as booting and testing
- Host resources
  - processor, sockets, cores
  - memory
  - disk
  - network interfaces
  - additional hardware devices, such as cryptographic and compression devices

Highly Automated Installation and Commissioning

HP Helion OpenStack Carrier Grade substantially simplifies and integrates the installation and commissioning sequence. Software for the initial controller node is installed from a USB flash drive or PXE boot server, and configured for operation with a single script. Software for subsequent hosts is installed over the internal management network from the initial controller node, and configured using either the web administration interface, or the command line on the controller.

This is in contrast with the installation and commissioning sequence of the open-source OpenStack, which involves the setup of numerous individual configuration files, the installation of various backends supporting OpenStack components, and the configuration of multiple Linux services and other open-source programs.

HP Helion OpenStack Carrier Grade also supports automated installation and commissioning using response files for initial configuration, and manifest files for bulk installation of subsequent hosts.

Performance Management Extensions

The Ceilometer service included with HP Helion OpenStack Carrier Grade features improved performance, scalability, and usability. It has been extended to support a CSV-formatted file backend that provides a more traditional Telco Northbound interface for performance management. HP Helion OpenStack Carrier Grade also adds performance monitoring for the Platform resource usage (CPU, memory, and disk), and for vswitch process.

Improved Web Administration Interface

HP Helion OpenStack Carrier Grade web administration interface is an enhanced version of the Horizon web interface provided by OpenStack. The enhancements include:

- Automatic page refresh for immediate notification of status changes. This eliminates the need for manual reloading of web pages to access up-to-date information about the cloud
- Admin Overview page with charts and tables providing a high-level overview of cloud resources
  - avg/max/min compute vCPU usage
  - avg/max/min compute memory usage
  - avg/max/min compute disk usage
  - avg/max/min AVS CPU utilization
  - avg/max/min provider network port utilization
  - current host status (available, unavailable, locked, unlocked)
- Improved performance management resource usage web page
  - optimized performance management sample DB queries for improved usability
  - capability to filter on fields in the performance management sample's metadata
  - human-readable legends and chart labels using object names instead of long UUID text strings
  - performance management meters can be selected using their brief descriptions
  - meters for CPU, memory, and disk utilization
- Ability to re-brand the Horizon GUI by modifying color schemes, logos, icons, and server naming.

Heat Extensions

The OpenStack Heat orchestration service is enhanced to include:

- Support for several resource types missing from the public reference implementation
- Support for multiple network interfaces at VM launch
- Simplified naming options for flavors and VM instances
- Enhanced server group support, including best-effort instantiation and group size limiting
- Simplified options for passing user data to an instance
- Improved stack access for users
- Support for dependencies during resource allocation
- Local-file or URL locations for Glance images
- Helpful example templates illustrating real reference scenarios

Backup and Restore

Tools and procedures to backup and restore system data, virtual machines, and storage resources.

Patching

Tools and procedures to patch system images to ensure they are always up to date with the latest release and security fixes.

Alarms Generation and Reporting

- alarms on performance management thresholds and cloud-level services and equipment
- SNMPv2c support

Supported OpenStack Services

HP Helion OpenStack Carrier Grade integrates the following OpenStack services into its cloud solution: Nova, Neutron, Keystone, Glance, Cinder, Horizon, Ceilometer, Heat.

REST API

HP Helion OpenStack Carrier Grade supports the following external REST APIs through its OAM floating IP address:

- OpenStack REST APIS
  - Block Storage API (with HP Helion OpenStack Carrier Grade extensions)
  - Compute API (with HP Helion OpenStack Carrier Grade extensions)
  - Identity API
  - Image Service API
  - Networking API (with HP Helion OpenStack Carrier Grade extensions)
  - Orchestration API
  - Telemetry API (with HP Helion OpenStack Carrier Grade extensions)
- HP Helion OpenStack Carrier Grade REST APIs
  - SysInv API
  - Patching API

Note:

HP Helion OpenStack Carrier Grade extensions use the OpenStack Extension Mechanism to ensure compatibility with existing clients.

HTTP or HTTPS protocol can be supported for these external REST APIs.

HP Helion OpenStack Carrier Grade SDK

The HP Helion OpenStack Carrier Grade Software Development Kit (SDK) provides drivers, daemons, API libraries, and configuration files that you can include in a guest image to leverage the extended capabilities of HP Helion OpenStack Carrier Grade. Available components include:

- Accelerated Kernel Network Drivers—Drivers for improved performance of kernel-based networking VNFs
- Accelerated DPDK Network Drivers—Drivers for high-performance DPDK-based networking VNFs
- VM Resource Scaling—A service for scaling the capacity of a guest server on demand
- Guest Heartbeat—A service for health monitoring of guest applications
- Server Group Messaging—A service for low-bandwidth peer messaging between servers in the same group
- SNMP MIB—Resources for system alarms management
- Sample Heat Orchestration Templates—Resources for deploying and managing stacks of applications or application services

- OpenStack REST API Documentation—Documentation for HP Helion OpenStack Carrier Grade REST APIs and HP Helion OpenStack Carrier Grade extensions to OpenStack REST APIs
- Configuration utilities—tools for generating and validating files used to automate HP Helion OpenStack Carrier Grade installation and configuration
- Custom Branding—Resources for customizing the Horizon GUI

For more information, see the *HP Helion OpenStack Carrier Grade Software Development Kit*.

Small Footprint Solution

HP Helion OpenStack Carrier Grade can be deployed on a small two-node redundant system where each node runs controller functions, storage functions, and compute functions, and hosts virtual machines. This provides an ideal solution for consoldiating a small set of standalone server-based products (possibly all running different operating systems, with different networking requirements, and so on) into a single 1 + 1 physical deployment solution, while at the same time leveraging all of the cloud benefits of easy-to-deploy, high-availability, autoscaling of resource usage, and so on.

OAM Network Firewall Rules

HP Helion OpenStack Carrier Grade provides an administrative command for performing a live packet trace capture, using tcpdump, on a logical interface of the virtual switch. The utility is intended to be used only in the development of Guest VM Applications or on non-roduction deployments, as it does result in non-performance degradation and potential packet loss.

Reference Logical Architecture

The HP Helion OpenStack Carrier Grade architecture supports various types of host, networks, and networking hardware in different configurations.

For more information on HP Helion OpenStack Carrier Grade architecture, see the *Technical Overview*.

Controller Nodes

Run the HP Helion OpenStack Carrier Grade services needed to manage the cloud infrastructure. The two controller nodes run the services in carrier grade mode, that is, as a high-availability cluster. See *Controller Nodes and High Availability* on page 138 for more information.

Controller nodes manage other hosts over the attached internal management network. They also provide administration interfaces to clients over the OAM network. Two controllers are necessary for HP Helion OpenStack Carrier Grade to operate properly.

Storage Node

Runs HP Helion OpenStack Carrier Grade storage services.

Storage servers are optional, but when available:

- two of them are mandatory for reliability purposes
- they must connect to both the internal management and infrastructure networks

Compute Node

Runs the HP Helion OpenStack Carrier Grade Server compute services and hosts the virtual machines.

A compute node connects to the controller nodes over the internal management network, and to the provider networks using its data interfaces.

Data Interfaces

In a compute node, a data interface is a logical network adapter used to connect to one or more provider networks. It is created by mapping a physical network interface on the compute node to the target provider networks. The mapping can use multiple physical network interfaces to support a LAG connection.

Provider networks used by a single data interface must share the same Ethernet MTU value.

Internal L2 Switch

An L2 switching facility, often implemented on a single Top-of-Rack (ToR) switch, used to realize the internal and infrastructure networks, and depending on the system configuration, the board management network. These networks are implemented as follows:

- The internal management network using a dedicated port-based VLAN. On ports connecting to controller nodes, VLAN tagging is enabled to connect them to the board management network also.
- The board management network as a port-based VLAN. This applies only if a board management network is in use, and is configured for internal access.
- The infrastructure network using a dedicated port-based VLAN.

Each host in the cluster connects to the internal L2 switch, as follows:

- One dedicated mother-board physical port to participate in the internal management network. This connection is mandatory.
  - Controller nodes connect also to the board management network using VLAN tagging on this same port.
- One dedicated physical port to participate in the infrastructure network. This connection is optional, but inclusive for all hosts. That is, if an infrastructure network is to be used, then all hosts must connect to it.
- Optionally, one physical iLO (Integrated Lights Out) port to participate in the board management network. This applies only if a board management network is in use, and is configured for internal access.

As long as the integrity and isolation of the internal, infrastructure, and board management networks are ensured, the internal L2 switch can be realized over physical switching resources that provide connectivity to other networks.

Internal Management Network

An isolated L2 network implemented on the internal L2 switch, used to enable communications among hosts for software installation, and management of hosts and virtual machines. This network is only accessible within the cluster, and for the most part, it is transparent to management operations of the cloud.

The internal management network must be unique and dedicated to the cluster. Sharing it with other cluster, or other non-related equipment, is not supported.

Infrastructure Network

An optional network used to improve overall performance for a variety of operational functions. When available, it is used by HP Helion OpenStack Carrier Grade during the following operations:

- control and data synchronization when migrating virtual machines between compute nodes
- isolation of storage traffic when accessing storage nodes providing storage services

Overall, an infrastructure network provides a target for the HP Helion OpenStack Carrier Grade to offload heavy traffic from the internal management network. This prevents sensitive traffic, such as internal heartbeat and monitoring messages, from being starved for bandwidth when background traffic, such as storage-related flows, peaks during normal operations.

When unavailable, all infrastructure traffic is carried over the internal management network.

OAM Network

A physical network used to provide access to the configuration and management facilities of HP Helion OpenStack Carrier Grade. It provides connectivity between the controller nodes and the edge router of the OAM network. The web administration interface, and the console interfaces (using SSH) to the controllers, are available on this network. Depending on the system configuration, the OAM network may also be used for controller access to the board management network.

The OAM network provides access to the OpenStack and HP Helion OpenStack Carrier Grade Server-specific REST APIs, which can be used by users and third-party developers to develop high-level cloud orchestration services.

The OAM network also provides the controller nodes with access to system-wide resources such as DNS and time servers. Access to the open Internet can also be provided if desired, at the discretion of each particular installation.

Board Management Network

An optional network used by the controller nodes to perform out-of-band reset and power-on/power-off operations on hosts equipped with iLO (Integrated Lights Out) board management modules.

This network can be configured for internal access or external access.

Internal access

In this configuration, the modules are accessible using a VLAN network implemented on an internal L2 switch.

External access

In this configuration, the modules are accessible using the OAM network.

Board management modules are optional. Associated maintenance operations are available only for hosts equipped with them.

For more information, refer to *The Board Management Network* on page 39.

Physical Network

A physical transport resource used to interconnect compute nodes among themselves and with external networks. Virtual networks, such as provider and tenant networks, are built on top of the physical network. Access to multiple physical networks can be defined by the HP Helion OpenStack Carrier Grade administrator.

Physical networks are not configured by the administrator. They are physically provisioned by the data center where the cluster is deployed.

Provider Network

A Layer 2 virtual network used to provide the underlying network connectivity needed to instantiate the tenant networks. Multiple provider networks may be configured as required, and realized over the same or different physical networks. Access to external networks is typically granted to the compute nodes via the provider network. The extent of this connectivity, including access to the open Internet, is application dependent.

Provider networks are created by the administrator to make use of an underlying set of resources on a physical network. They can be created as being of one of the following types:

flat

A provider network mapped entirely over the physical network. The physical network is used as a single Layer 2 broadcast domain. Each physical network can realize at most one flat provider network.

A provider network of this type supports at most one tenant network, even if its corresponding shared flag is enabled.

VLAN

A provider network implemented over a range of IEEE 802.1Q VLAN identifiers supported by the physical network. This allows for multiple provider networks to be defined over the same physical network, all operating over non-overlapping sets of VLAN IDs.

A set of consecutive VLAN IDs over which the provider network is defined is referred to as a network's **segmentation range**. A provider network can have more than one segmentation range. Each VLAN ID in a segmentation range is used to support the implementation of a single tenant network.

A segmentation range can be shared by multiple tenants if its **shared** flag is set. Otherwise, the segmentation range supports tenant networks belonging to a single specified tenant.

VXLAN

A provider network implemented over a range of VXLAN Network Identifiers (VNIs.) This is similar to the VLAN option, in that it allows multiple provider networks to be defined over the same physical network using unique VNIs defined in segmentation ranges. In addition, VXLAN provides a Layer 2 overlay scheme on Layer 3 networks, enabling connectivity between Layer 2 segments separated by one or more Layer 3 routers.

Tenant Network

A virtual network associated with a tenant. A tenant network is instantiated on a compute node, and makes use of a provider network, either directly over a flat network, or using technologies such as VLAN and VXLAN.

Tenant networks use the high-performance virtual L2 switching capabilities built into the HP Helion OpenStack Carrier Grade stack of the compute node. They provide switching facilities to the virtual service instances, to communicate with external resources and with other virtual service instances running on the same or different compute nodes.

When instantiated over a provider network of the VLAN or VXLAN type, the VLAN ID or VNI for the tenant network is assigned automatically. The allocation algorithm selects the lowest available ID from any segmentation range owned by the tenant. If no such ID is available, it selects the lowest available ID from any shared segmentation range. The system reports an error when no available ID is found.

Tenant networks created by the administration can also be configured to use a pre-selected VLAN ID or VNI. This can be used to extend connectivity to external networks.

Controller Floating IP Address

A unique IP address shared by the cluster of controller nodes. Only the master controller node is active on this address at any time. The IP address is floating in the sense that it is automatically transferred from one controller node to the other, as dictated by the high-availability directives of the cluster.

The controller cluster has two floating IP addresses, one facing the OAM network, only seen by the web interface SSH administration clients and REST APIs, and another facing the internal management network, only seen by the compute nodes.

Link Aggregation (LAG) or Aggregated Ethernet (AE)

A mechanism that allows multiple parallel Ethernet network connections between two hosts to be used as a single logical connection. HP Helion OpenStack Carrier Grade supports LAG connections on all its Ethernet connections for the purpose of protection (fault-tolerance) only. Different modes of operation are supported.

OAM Network L2 Switch(es)

One or more switches used to provide an entry point into the OAM network.

Note:

The OAM ports on the controller do not support VLAN tagging. If needed, port- or MAC-based VLANs defined on the L2 switch can be used to steer OAM traffic appropriately.

Provider Network L2 Switches

Provide the entry point into the provider networks. It is through these L2 switches that the compute nodes get integrated as active end points into each of the provider networks.

Edge Router

Provides connectivity to external networks as needed by controller and compute nodes.

Reachability to the open Internet is not mandatory, but strictly application-dependent. Guest applications running on the compute nodes may need to access, or be reachable from, the Internet. Additionally, access to the OAM Network from external networks might be desirable.

Web Administration Interface

Provides the HP Helion OpenStack Carrier Grade main management interface. It is available using any W3C standards-compliant web browser, from any point within the OAM Network where the OAM floating IP address is reachable.

About Tenants (Projects) and Users

Tenants and users are system resources managed by the OpenStack Keystone service.

Tenants are the core resource structure on which all end user services are managed. They are isolated resource containers consisting of networks, storage volumes, images, virtual machines, authentication keys, and users.

When HP Helion OpenStack Carrier Grade is deployed, two default tenants are created: **admin** and **services**. They are used to group resources to be associated with the user **admin** and the cloud services, respectively.

Note:

Earlier versions of OpenStack used the term *project* instead of *tenant*. Because of this legacy terminology, the web administration interface uses both terms, and some command-line tools use **--project_id** when a tenant ID is expected.

Users are system resources that can operate on one or more tenants, within the constraints of a particular role. For each user, the Keystone service maintains a list of (tenant, role) tuples, which are used to determine the tenants the user can operate on, and the role the user should play on each of them.

In the default installation of HP Helion OpenStack Carrier Grade, several users are already defined. Each of the OpenStack services, such as Nova and Neutron, exist as system users. They all operate on the default tenant **services**.

The Keystone **admin** user is associated with the **admin** tenant. This user has administrator privileges for creating, modifying, and deleting OpenStack resources, including creating other tenants and users.

Linux User Accounts

Linux user accounts are available on all hosts for administration, operation, and general hosting purposes. They have no relation with the cloud user tenant accounts created using the web management interface, the CLI commands, or the APIs.

You can log in remotely as a Linux user by using an SSH connection and specifying the OAM floating IP address as the target. This establishes a connection to the currently active controller. If the OAM floating IP address moves from one controller node to another, the SSH session is blocked. To ensure access to a particular controller regardless of its current role, specify the controller physical address instead.

The wrsroot Account

This is a local, per-host, account created automatically when a new host is provisioned. On controller nodes, this account is available even before the **config_controller** script is executed.

The default initial password is **wrsroot**.

- The initial password must be changed immediately when you log in to **controller-0** for the first time. For details, see the *HP Helion OpenStack Carrier Grade Software Installation Guide*.
- After five consecutive unsuccessful login attempts, further attempts are blocked for about five minutes.

Subsequent password changes must be executed on the active controller to ensure that they propagate to all other hosts in the cluster. Otherwise, they remain local to the host where they were executed, and are overwritten on the next reboot to match the password on the active controller.

From the **wrsroot** account, you can execute commands requiring different privileges.

- You can execute non-root level commands as a regular Linux user directly.

  If you do not have sufficient privileges to execute a command as a regular Linux user, you may receive a permissions error, or in some cases, the command may be reported as not found.
- You can execute root-level commands as the **root** user.

  To become the root user, use the `sudo` command to elevate your privileges, followed by the command to be executed.

  If a password is requested, provide the password for the **wrsroot** account.
- You can execute OpenStack administrative commands as the Keystone **admin** user.

  Source the script `/etc/nova/openrc` while working on the active controller to acquire Keystone administrative privileges.

  ```
  $ source /etc/nova/openrc
  [wrsroot@controller-0 ~(keystone_admin)]$
  ```

  The system prompt changes to indicate the new acquired privileges.

  Note:

  The default Keystone prompt includes the host name and the current working path. For simplicity, this guide uses the following generic prompt instead:

  ```
  ~(keystone_admin)$
  ```

  For more information on the active controller, see *Controller Nodes and High Availability* on page 138.

Local User Accounts

You can manage regular Linux user accounts on any host in the cluster using standard Linux commands. New accounts created on one host are not automatically propagated to other hosts.

Password changes are not enforced automatically on first login, and they are not propagated by the system (with the exception of the **wrsroot** account, for which passwords changed on the active controller are propagated to other hosts.) Any special considerations for these accounts, if any, must be configured manually.

Local user accounts can be added to the *sudoers* list using the `visudo` command. They can also source the script `/etc/nova/openrc` to become OpenStack administrators when working on the active controller.

Backup and restore operations of home directories and passwords must be done manually. They are ignored by the system backup and restore utilities. See *System Backups* on page 206 for further details.

LDAP User Accounts

You can create regular Linux user accounts using the HP Helion OpenStack Carrier Grade LDAP service. LDAP accounts are centrally managed; changes made on any host are propagated automatically to all hosts on the cluster.

In other respects, LDAP user accounts behave as any local user account. They can be added to the sudoers list, and can acquire OpenStack administration credentials when executing on the active controller.

LDAP user accounts share the following set of attributes:

- The initial password is the name of the account.
- The initial password must be changed immediately upon first login.
- Requirements for new passwords include:
  - to be at least eight characters long
  - to have at least one lowercase character
  - to differ in at least three characters from the previous password
  - not to be evidently trivial to guess, such as a2345678, or a reversed version of the old password
- Login sessions are logged out automatically after about 15 minutes of inactivity.
- The accounts block following five consecutive unsuccessful login attempts. They unblock automatically after a period of about five minutes.
- Home directories are created dynamically on first login. Note that home directories for local user accounts are created when the accounts are created.
- All authentication attempts are recorded on the file `/var/log/auth.log` of the target host.
- Home directories and passwords are backed up and restored by the system backup utilities.

The following LDAP user accounts are available by default on newly deployed hosts, regardless of their personality:

admin

A cloud administrative account, comparable in purpose to the default **admin** account used in the web management interface.

This user account operates on a restricted Linux shell, with very limited access to native Linux commands. However, the shell is preconfigured to have administrative access to OpenStack commands, including the available HP Helion OpenStack Carrier Grade CLI extensions.

operator

A host administrative account. It has access to all native Linux commands and is included in the sudoers list.

For increased security, the **admin** and **operator** accounts must be used from the console ports of the hosts; no SSH access is allowed.

Managing LDAP User Accounts

Although the scope of operations for the LDAP user accounts is local, that is, they operate on the target host exclusively, management of these accounts operates at the cluster level. This means that operations such as password

change, and addition or removal of users, are applied to the entire cluster. For example, a password change executed while logged into controller-0, is effective immediately on all other hosts in the cluster.

Centralized management is implemented using two LDAP servers, one running on each controller node. LDAP server synchronization is automatic using the native LDAP content synchronization protocol.

A set of LDAP commands is available to operate on LDAP user accounts. The commands are installed in the directory `/usr/local/sbin`, and are available to any user account in the sudoers list. Included commands are `lsldap`, `ldapadduser`, `ldapdeleteuser`, and several others starting with the prefix `ldap`. Use the command option --help on any command to display a brief help message, as illustrated below.

```
$ ldapadduser --help
Usage : /usr/local/sbin/ldapadduser <username> <groupname | gid> [uid]
$ ldapdeleteuser --help
Usage : /usr/local/sbin/ldapdeleteuser <username | uid>
```

Network Requirements

The HP Helion OpenStack Carrier Grade networking environment incorporates up to five types of network: the internal management network, the OAM network, one or more provider networks, an optional infrastructure network, and an optional board management network. Operational requirements for each network are described in the following sections.

Note:

The HP Helion OpenStack Carrier Grade controllers use IP multicast messaging on the management, infrastructure, and OAM networks. To prevent loss of controller synchronization, ensure that the switches and other devices on these networks are configured with appropriate settings.

Internal Management Network

The internal management network must be implemented as a single, dedicated, Layer 2 broadcast domain for the exclusive use of each HP Helion OpenStack Carrier Grade cluster. Sharing of this network by more than one HP Helion OpenStack Carrier Grade cluster is not a supported configuration.

During the HP Helion OpenStack Carrier Grade software installation process, several network services such as BOOTP, DHCP, and PXE, are expected to run over the internal management network. These services are used to bring up the different hosts to an operational state. Therefore, it is mandatory that this network be operational and available in advance, to ensure a successful installation.

On each host, the internal management network can be implemented using a 1Gb or 10 Gb Ethernet port. In either case, requirements for this port are:

- it must be capable of PXE-booting
- can be used by the motherboard as a primary boot device

Infrastructure Network

This is an optional network. As with the internal management network, the infrastructure network must be implemented as a single, dedicated, Layer 2 broadcast domain for the exclusive use of each HP Helion OpenStack Carrier Grade cluster. Sharing of this network by more than one HP Helion OpenStack Carrier Grade cluster is not a supported configuration.

The infrastructure network can be implemented as a 1Gb or 10 Gb Ethernet network. In its absence, all infrastructure traffic is carried over the internal management network.

OAM Network

You should ensure that the following services are available on the OAM Network:

DNS Service

Needed to facilitate the name resolution of servers reachable on the OAM Network.

HP Helion OpenStack Carrier Grade can operate without a configured DNS service. However, a DNS service should be in place to ensure that links to external references in the current and future versions of the web administration interface work as expected.

NTP Service

The Network Time Protocol (NTP) can be optionally used by the HP Helion OpenStack Carrier Grade controller nodes to synchronize their local clocks with a reliable external time reference. However, it is strongly suggested that this service be available, among other things, to ensure that system-wide log reports present a unified view of the day-to-day operations.

The HP Helion OpenStack Carrier Grade compute nodes always use the controller nodes as the de-facto time server for the entire HP Helion OpenStack Carrier Grade cluster.

Provider Network

There are no specific requirements for network services to be available on the provider network. However, you must ensure that all network services required by the guests running in the compute nodes are available. For configuration purposes, the compute nodes themselves are entirely served by the services provided by the controller nodes over the internal management network.

Board Management Network

The board management network implementation depends on the system configuration.

Internal access

On a system configured for internal access to the board management network, the board management modules are connected to VLAN-tagged ports on the internal L2 switch, and the controller is attached to the VLAN using the management network interface. For more about the internal L2 switch, refer to *Reference Logical Architecture* on page 14.

External access

On a system configured for external access to the board management network, the board management modules are assigned IP addresses accessible from the OAM network, and the controller uses the OAM network to connect to them.

The system configuration is determined at installation. If a board management network is specified during the controller configuration script, then the network is configured for internal access. Otherwise, the board managament network uses external access. For more information, see *The Board Management Network* on page 39.

Dashboard Controls

The HP Helion OpenStack Carrier Grade web administration interface provides a dashboard for managing all aspects of the system.

See the HP Helion OpenStack Carrier Grade 2.0 Beta: User Guide.

Chapter 2

Configuration Planning

Topics:

System configuration options are available when installing the HP Helion OpenStack Carrier Grade software. You should prepare a set of selected values and features ready to use which are applicable to a specific deployment scenario.

Ethernet Interfaces

Ethernet interfaces, both physical and virtual, play a key role in the overall performance of the virtualized network. Therefore, it is important to understand the available interface types, their configuration options, and their impact on network design.

On compute node data interfaces, HP Helion OpenStack Carrier Grade supports Ethernet Network Interface Cards (NIC) based on the following chip sets:

- Intel I350 (Powerville) 1G
- Intel 82599 (Niantic) 10 G
- Mellanox Technologies MT27500 Family [ConnectX-3] 10G / 40G

The following virtual network interfaces are available:

- avp (Accelerated Virtual Port)
- e1000 (Intel e1000 Emulation)
- ne2k_pci (NE2000 Emulation)
- pcnet (AMD PCnet/PCI Emulation)
- rtl8139 (Realtek 8139 Emulation)
- virtio (VirtIO Network)
- pci-passthrough (PCI Passthrough Device)
- pci-sriov (SR-IOV device)

About LAG/AE Interfaces

To provide link protection, Ethernet interfaces in a LAG group can be attached either to the same L2 switch, or to multiple switches in a redundant configuration. For more information about L2 switch configurations, see *L2 Access Switches* on page 59. For information about the different LAG modes, see the *HP Helion OpenStack Carrier Grade Software Installation Guide*.

Ethernet Interface Configuration

You can review and modify the configuration for physical or virtual Ethernet interfaces using the web administration interface or the CLI.

Physical Ethernet Interfaces

The physical Ethernet interfaces on HP Helion OpenStack Carrier Grade nodes are configured to use the following networks:

- the internal management network
- the external OAM network
- the infrastructure network, if present
- one or more data networks

A single interface can optionally be configured to support more than one network using VLAN tagging (see *Shared (VLAN) Ethernet Interfaces* on page 24). In addition, the management or infrastructure interfaces, or both, can be configured with an additional data network (see the *HP Helion OpenStack Carrier Grade Software Installation Guide: Editing Interface Settings*).

On the controller nodes, all Ethernet interfaces are configured automatically when the nodes are initialized, based on the information provided in the controller configuration script (see the *HP Helion OpenStack Carrier Grade Installation Guide: The Controller Configuration Script*). On compute nodes, the Ethernet interfaces for the management network are configured automatically. The remaining interfaces require manual configuration. Interface configurations are summarized in the following table.

Note:

If a network attachment uses LAG, the corresponding interfaces on the storage and compute nodes must also be configured manually to specify the interface type.

Table 1: Interface Configuration by Network Type

	Controller	Compute
Mgmt	Configured automatically	Configured automatically
OAM	Configured automatically	Not used
Infra	Configured automatically	Configured manually
Data	Not used	Configured manually

You can review and modify physical interface configurations from the web administration interface or the CLI. For more information, see the *HP Helion OpenStack Carrier Grade Installation Guide: Editing Interface Settings*, the *HP Helion OpenStack Carrier Grade Installation Guide: Creating Data Interfaces*, and the *HP Helion OpenStack Carrier Grade Installation Guide: Command-line Installation*.

You can also save the interface configurations for a particular node to use as a *profile* or template when setting up other nodes. For more information, see *Interfaces* on page 131.

Virtual Ethernet Interfaces

The virtual Ethernet interfaces for guest VMs running on HP Helion OpenStack Carrier Grade are defined when an instance is launched. They connect the VM to *tenant networks*, which are virtual networks defined over *provider networks*, which in turn are abstractions associated with physical interfaces assigned to data networks on the compute nodes. Several virtual interface driver options are available. For more information about launching instances and connecting their virtual Ethernet interfaces, see the *HP Helion OpenStack Carrier Grade Reference Deployment Scenarios*. The chapters on *Deploying the Bridging Scenario* and *Deploying the Routing Scenario* contain detailed examples for defining virtual Ethernet interfaces.

You can also connect a VM directly to a physical interface using PCI passthrough (see *PCI Passthrough Ethernet Interfaces* on page 29) or SR-IOV (see *SR-IOV Ethernet Interfaces* on page 33).

Shared (VLAN) Ethernet Interfaces

The management, OAM, infrastructure, and data networks can share Ethernet or aggregated Ethernet interfaces using VLAN tagging.

As explained in *Network Planning* on page 36, the internal management network cannot use VLAN tagging. However, the OAM, infrastructure, and data networks can use VLAN tagging, allowing them to share an Ethernet or aggregated Ethernet interface with other networks.

Note:

You cannot configure a data VLAN or add a data network on an aggregated Ethernet interface.

For a system using all four networks, the following arrangements are possible:

- One interface for the management network, another interface for the OAM network, a third for the infrastructure network, and one or more additional interfaces for data networks.
- One interface for the management network, and a second interface for either the OAM or infrastructure network, with the remaining networks implemented using VLAN tagging on either interface.
- One interface for the management network, and a second carrying the OAM and infrastructure networks, both implemented using VLAN tagging, with data networks implemented on either or both interfaces using VLAN tagging.

- One interface for the management network, with the OAM, infrastructure, and data networks also implemented on it using VLAN tagging.

Note:

Data networks implemented using VLAN tagging are not compatible with VLAN-based provider networks. (Stacked VLANs are not supported.). To support VLAN provider networks, you can configure a data network on a management or infrastructure interface by editing the interface and selecting both types of network, and then selecting the VLAN provider network. For more information, see the Network Type discussion in the *HP Helion OpenStack Carrier Grade Installation Guide: Interface Settings*.

Options to share an interface using VLAN tagging are presented during the configuration controller script (see the *HP Helion OpenStack Carrier Grade Installation Guide: The Controller Configuration Script*). To attach an interface to other networks after configuration, you can edit the interface; for details, see the *HP Helion OpenStack Carrier Grade Installation Guide: Attaching to Networks Using a VLAN Interface*.

You can also use the command line by logging into the active controller and becoming the Keystone **admin** user, and then using a command such as the following for each node:

```
~(keystone_admin)$ system host-if-add controller-0 -V 22 -nt infra infra0
 vlan none eth1
+------------------+--------------------------------------+
| Property         | Value                                |
+------------------+--------------------------------------+
| ifname           | infra0                               |
| networktype      | infra                                |
| iftype           | vlan                                 |
| ports            | []                                   |
| providernetworks | None                                 |
| imac             | 08:00:27:f2:0d:68                    |
| imtu             | 1500                                 |
| aemode           | None                                 |
| schedpolicy      | None                                 |
| txhashpolicy     | None                                 |
| uuid             | 8ca9854e-a18e-4a3c-8afe-f050da702fdf |
| ihost_uuid       | 3d207384-7d30-4bc0-affe-d68ab6a00a5b |
| vlan_id          | 22                                   |
| uses             | [u'eth1']                            |
| used_by          | []                                   |
| created_at       | 2015-02-04T16:23:28.917084+00:00     |
| updated_at       | None                                 |
+------------------+--------------------------------------+
```

This example configures the **eth1** interface on **controller-0** to connect to the infrastructure network using VLAN ID 22. (The **none** parameter is used in place of a provider network name, indicating no provider network.)

To display VLAN information as well as dependencies, use a command such as the following:

```
~(keystone_admin)$ system host-if-list controller-0
...+-------+------...+----------+---------+-----------+------------
+-------------+...
...| name   | netwo...| type     | vlan id | ports     | uses i/f   | used
 by i/f |...
...+-------+------...+----------+---------+-----------+------------
+-------------+...
...| infra0 | infra...| vlan     | 22      | []        | [u'mgmt0'] | []
      |...
...| oam0   | oam  ...| ethernet | None    | [u'eth0'] | []         | []
      |...
...| mgmt0  | mgmt ...| ethernet | None    | [u'eth1'] | []         |
 [u'infra0'] |...
```

```
...+--------+------...+----------+---------+-----------+------------
+-------------+...
```

The output lists all interfaces and associated networks. For each network that uses VLAN tagging, the **vlan id** is shown, as well as the shared interface used (in the **uses i/f** column). For shared interfaces, the names of the networks that use VLAN tagging on the interface are shown (in the **used by i/f** column).

For a LAG interface, the dependencies are shown as follows.

```
...+--------+------...+----------+---------+-----------+------------
+-------------+...
...| name   | netwo...| type     | vlan id | ports     | uses i/f   | used
 by i/f |...
...+--------+------...+----------+---------+-----------+------------
+-------------+...
...| oam0   | oam  ...| ethernet | None    | [u'eth0'] | []         | []
      |...
...| bond0  | mgmt ...| ae       | None    | []        | [u'eth1']  | []
      |...
...| eth1   | None ...| ethernet | None    | [u'eth1'] | []         |
 [u'bond0']  |...
...| infra0 | infra...| ethernet | None    | [u'eth3'] | []         | []
      |...
...+--------+------...+----------+---------+-----------+------------
+-------------+...
```

To see all available interfaces, add the -a flag.

```
~(keystone_admin)$ system host-if-list -a controller-0
...+--------+------...+----------+---------+-----------+------------
+-------------+...
...| name   | netwo...| type     | vlan id | ports     | uses i/f   | used
 by i/f |...
...+--------+------...+----------+---------+-----------+------------
+-------------+...
...| eth3   | None ...| ethernet | None    | [u'eth3'] | []         | []
      |...
...| infra0 | infra...| vlan     | 22      | []        | [u'mgmt0'] | []
      |...
...| eth2   | None ...| ethernet | None    | [u'eth2'] | []         | []
      |...
...| oam0   | oam  ...| ethernet | None    | [u'eth0'] | []         | []
      |...
...| mgmt0  | mgmt ...| ethernet | None    | [u'eth1'] | []         |
 [u'infra0'] |...
...+--------+------...+----------+---------+-----------+------------
+-------------+...
```

The Ethernet MTU

The Maximum Transmission Unit (MTU) of an Ethernet frame is a configurable attribute in HP Helion OpenStack Carrier Grade. Changing its default size must be done in coordination with other network elements on the Ethernet link.

In the context of HP Helion OpenStack Carrier Grade, the Maximum Transmission Unit (MTU) refers to the largest possible payload on the Ethernet frame on a particular network link. The payload is enclosed by the Ethernet header (14 bytes) and the CRC (4 bytes), resulting in an Ethernet frame that is 18 bytes longer than the MTU size.

The original IEEE 802.3 specification defines a valid standard Ethernet frame size to be from 64 to 1518 bytes, accommodating payloads ranging in size from 46 to 1500 bytes. Ethernet frames with a payload larger than 1500 bytes are considered to be jumbo frames.

For a VLAN network, the frame also includes a 4-byte VLAN ID header, resulting in a frame size 22 bytes longer than the MTU size.

For a VXLAN network, the frame is either 54 or 74 bytes longer, depending on whether IPv4 or IPv6 protocol is used. This is because, in addition to the Ethernet header and CRC, the payload is enclosed by an IP header (20 bytes for Ipv4 or 40 bytes for IPv6), a UDP header (8 bytes), and a VXLAN header (8 bytes).

In HP Helion OpenStack Carrier Grade, you can configure the MTU size for the following interfaces and networks:

- The management, OAM, and infrastructure network interfaces on the controller. The MTU size for these interfaces is set during initial installation. You can update the infrastructure interface MTU for the controller from the web administration interface or the CLI.
- Data interfaces on compute nodes. For more information, see the *Compute Node* entry in *Reference Logical Architecture* on page 14.
- Provider networks. For more information, see the *Provider Networks* entry in *Reference Logical Architecture* on page 14.

In all cases, the default MTU size is 1500. The minimum value is 576, and the maximum is 9216.

Since data interfaces are defined over physical interfaces connecting to provider networks, it is important to consider the implications of modifying the default MTU size:

- The MTU sizes for a data interface and the corresponding Ethernet interface on the edge router or switch must be compatible. You must ensure that each side of the link is configured to accept the maximum frame size that can be delivered from the other side. For example, if the data interface is configured with a MTU size of 9216 bytes, the corresponding switch interface must be configured to accept a maximum frame size of 9238 bytes, assuming a VLAN tag is present.

  The way switch interfaces are configured varies from one switch manufacturer to another. In some cases you configure the MTU size directly, while in some others you configure the maximum Ethernet frame size instead. In the latter case, it is often unclear whether the frame size includes VLAN headers or not. In any case, you must ensure that both sides are configured to accept the expected maximum frame sizes.
- For a VXLAN network, the additional IP, UDP, and VXLAN headers are invisible to the data interface, which expects a frame only 18 bytes larger than the MTU. To accommodate the larger frames on a VXLAN network, you must specify a larger nominal MTU on the data interface. For simplicity, and to avoid issues with stacked VLAN tagging, some third party vendors recommend rounding up by an additional 100 bytes for calculation purposes. For example, to attach to a VXLAN provider network with an MTU of 1500, a data interface with an MTU of 1600 is recommended.
- A provider network can only be associated with a compute node data interface with an MTU of equal or greater value.
- The MTU size of a compute node data interface cannot be modified to be less than the MTU size of any of its associated provider networks.
- The MTU size of a provider network can be modified only when there are no tenant networks that depend on the provider network.
- The MTU size of a provider network is automatically propagated up to any derived tenant networks.
- The Neutron L3 and DHCP agents automatically propagate the MTU size of their networks to their Linux network interfaces.
- The Neutron DHCP agent makes the option interface-mtu available to any DHCP client request from a virtual machine. The request response from the server is the current interface's MTU size, which can then be used by the client to adjust its own interface MTU size.
- The AVS prevents any AVP-Kernel or AVP-DPDK instances from setting a link MTU size that exceeds the maximum allowed on the corresponding tenant network. No such verification is available for virtio VM instances.

You can configure the MTU size from the web management interface when creating or modifying data interfaces and provider networks. On the CLI, use the option --mtu to modify the MTU size. For example:

```
~(keystone_admin)$ neutron providernet-create common-net --type vlan --mtu
 9216
```

This command creates a provider network named **common-net** with an MTU size of 9216 bytes.

Address Filtering on Virtual Interfaces

The AVS on compute nodes can be configured to filter out packets based on source MAC address.

MAC addresses for virtual network interfaces on virtual machines are dynamically allocated by the system. For most scenarios, the assigned MAC addresses are expected to be used on all outgoing packets from the virtual machine instances. However, there are scenarios where the source MAC address is not expected to match the original assignment, such as when a L2 switch is implemented internally on the virtual machine.

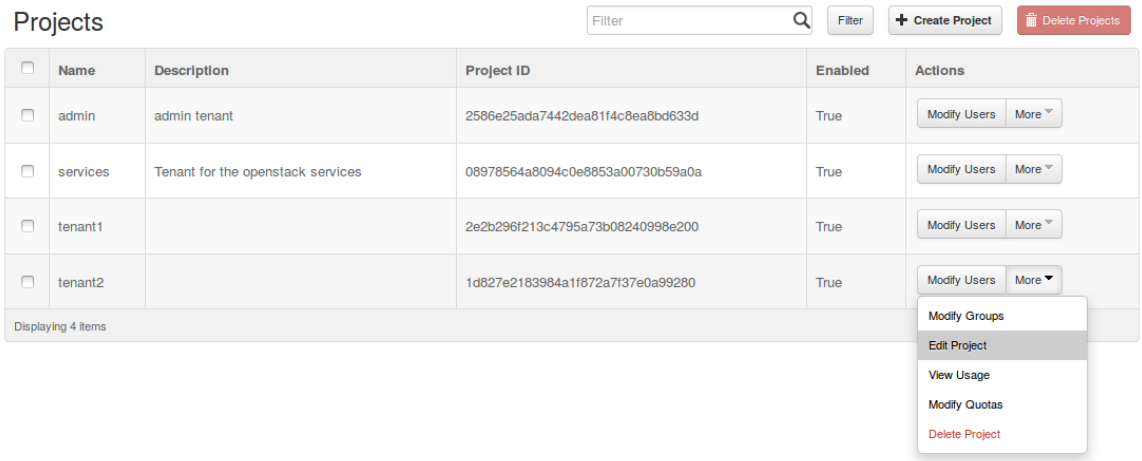
By default, the AVS on compute nodes accepts any source MAC address on the attached virtual network interfaces. However, it can be configured to filter out all incoming packets with non-system-generated source MAC address, if required. When evaluating the use of the filtering capability, you must consider the following:

- Source MAC address filtering can be enabled and disabled by the administrator user only, not by tenants.
- Filtering is enabled on a per-tenant basis only. Higher granularity, such as per-instance filtering, is not supported.
- When enabled, source MAC address filtering applies to all new virtual interfaces created by the Neutron service. Address filtering is not active on virtual interfaces created before filtering is enabled.

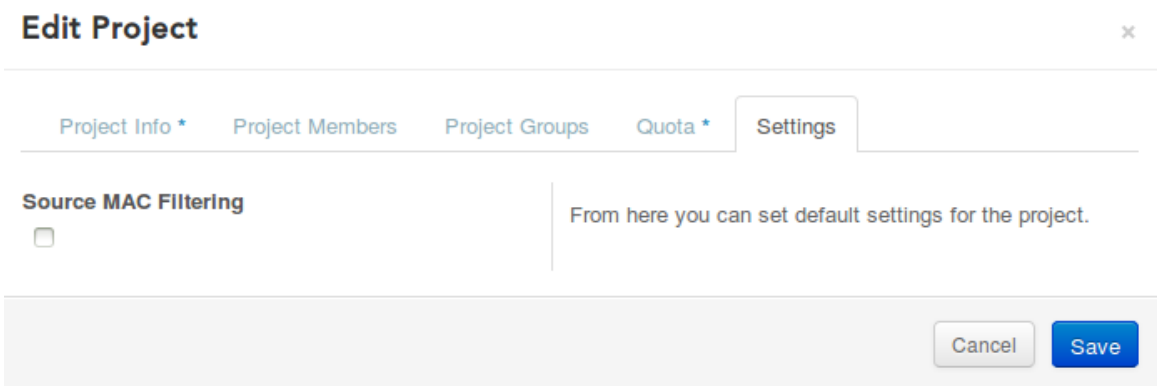
Use the following command to enable source MAC address filtering:

```
~(keystone_admin)$ neutron setting-update --tenant-id=<TENANTID> \
--mac-filtering={True|False}
```

Filtering can be enabled or disabled from the web management interface. As the administrator, select **Projects** from the **Identity** menu to display the **Projects** window. Then select the option Edit Project from the More drop-down menu of the desired tenant, as illustrated below.



The **Edit Project** window is displayed. The filtering option is available from the tab **Settings**, as illustrated below.



PCI Passthrough Ethernet Interfaces

A *passthrough* Ethernet interface is a physical PCI Ethernet NIC on a compute node to which a virtual machine is granted direct access. This minimizes packet processing delays but at the same time demands special operational considerations.

For all purposes, a PCI passthrough interface behaves as if it were physically attached to the virtual machine. Therefore any potential throughput limitations coming from the virtualized environment, such as the ones introduced by internal copying of data buffers, are eliminated. However, by bypassing the virtualized environment, the use of PCI passthrough Ethernet devices introduces several restrictions that must be taken into consideration. They include:

- no support for LAG, QoS, ACL, or host interface monitoring
- no support for live migration
- no access to the compute node's AVS switch

A passthrough interface bypasses the compute node's AVS switch completely, and is attached instead directly to the provider network's access switch. Therefore, proper routing of traffic to connect the passthrough interface to a particular tenant network depends entirely on the VLAN tagging options configured on both the passthrough interface and the access port on the switch.

The access switch routes incoming traffic based on a VLAN ID, which ultimately determines the tenant network to which the traffic belongs. The VLAN ID is either explicit, as found in incoming tagged packets, or implicit, as defined by the access port's default VLAN ID when the incoming packets are untagged. In both cases the access switch must be configured to process the proper VLAN ID, which therefore has to be known in advance.

To review the current PCI passthrough and SR-IOV device assignments for a given provider network, see *Displaying Provider Network Information* on page 42.

In the following example a new virtual machine is launched by user **user1** on tenant **tenant1**, with a passthrough interface connected to the tenant network **net0** identified with VLAN ID 10.

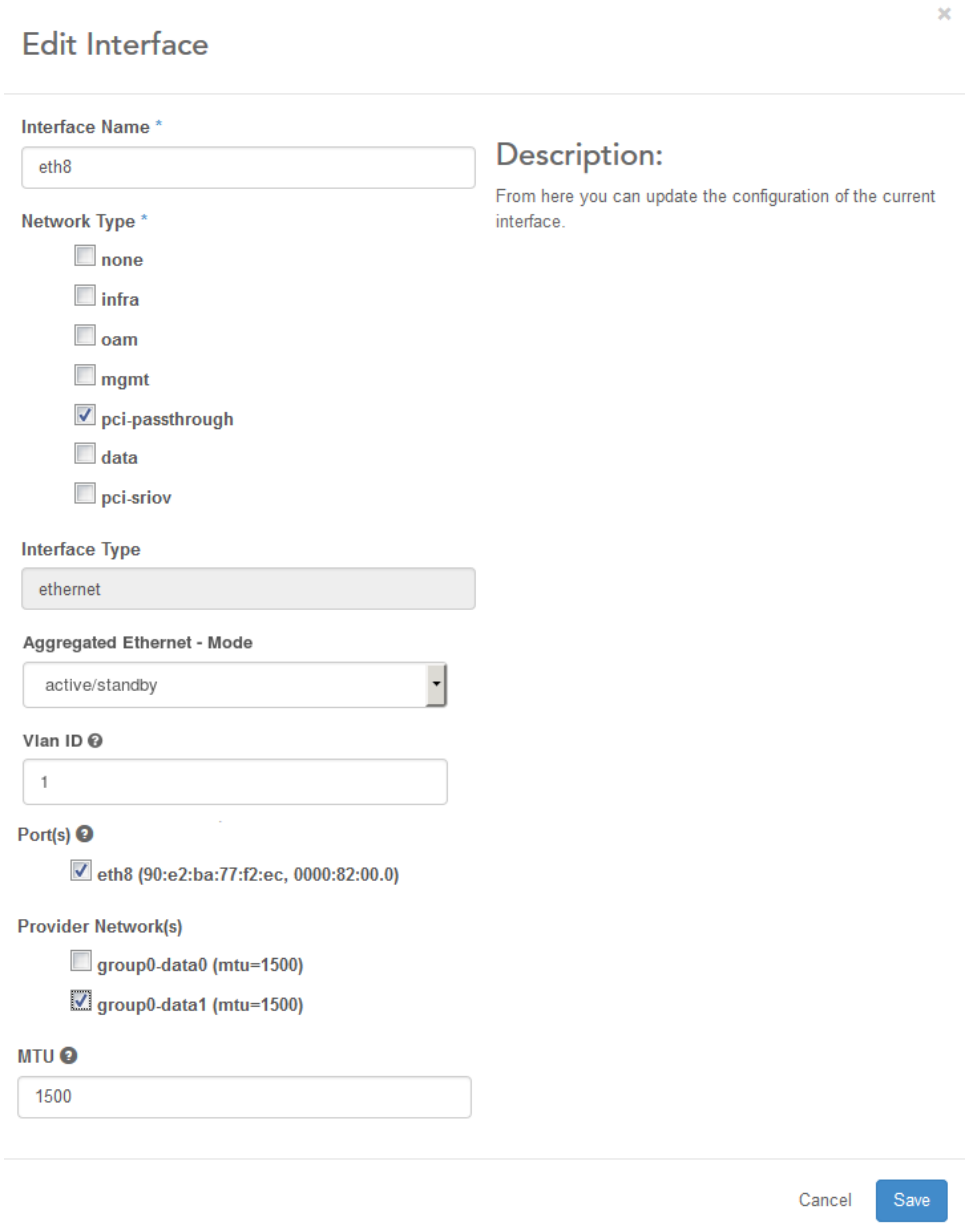
Note:

To use PCI passthrough or SR-IOV interfaces, you must have Intel VT-d features enabled in the BIOS.

The exercise assumes that the underlying provider network **group0-data0** exists already, and that VLAN ID 10 is a valid segmentation ID assigned to **tenant1**.

1. Log in as the **admin** user to the web management interface.
2. Lock the compute node you want to configure.
3. Configure the Ethernet interface to be used as a PCI passthrough interface.

   Select the **System Inventory** window from the **Admin** panel, click the **Hosts** tab, click the name of the compute node where the PCI interface is available, click the **Interfaces** tab, and finally click the **Edit Interface** button associated with the interface you want to configure. Fill in the window as illustrated below:



In this example, Ethernet interface **eth8** is assigned the network type *pci-passthrough*, and configured as connected to provider network **group0-data0**. Click the **Save** button to proceed.

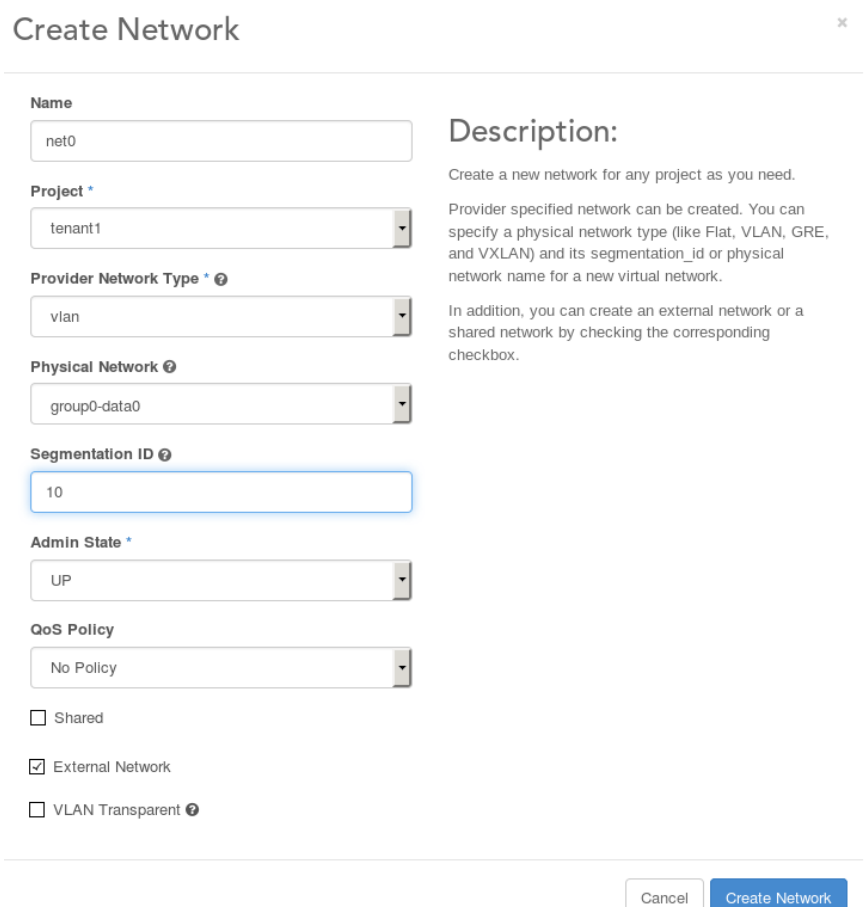
The interface can also be configured from the CLI as illustrated below:

```
~(keystone_admin)$ system host-if-modify \
-nt pci-passthrough -p group0-data0 compute-0 eth8
```

4. Create the **net0** tenant network

   Select the **Networks** window from the **Admin** panel, click the **Networks** tab, and then click the **Create Network** button. Fill in the **Create Network** window as illustrated below. You must ensure that:

   - The **tenant1** tenant has access to the tenant network, either assigning it as the owner, as in the illustration (using **Project**), or by enabling the shared flag.
   - The segmentation ID is set to 10.



Click the **Create Network** button to proceed.

5. Configure the access switch.

   Configure the physical port on the access switch used to connect to Ethernet interface **eth8** as an access port with default VLAN ID of 10. Traffic across the connection is therefore untagged, and effectively integrated into the targeted tenant network.

   You can also use a trunk port on the access switch so that it handles tagged packets as well. However, this opens the possibility for guest applications to join other tenant networks using tagged packets with different VLAN IDs, which might compromise the security of the system. See *L2 Access Switches* on page 59 for other details regarding the configuration of the access switch.

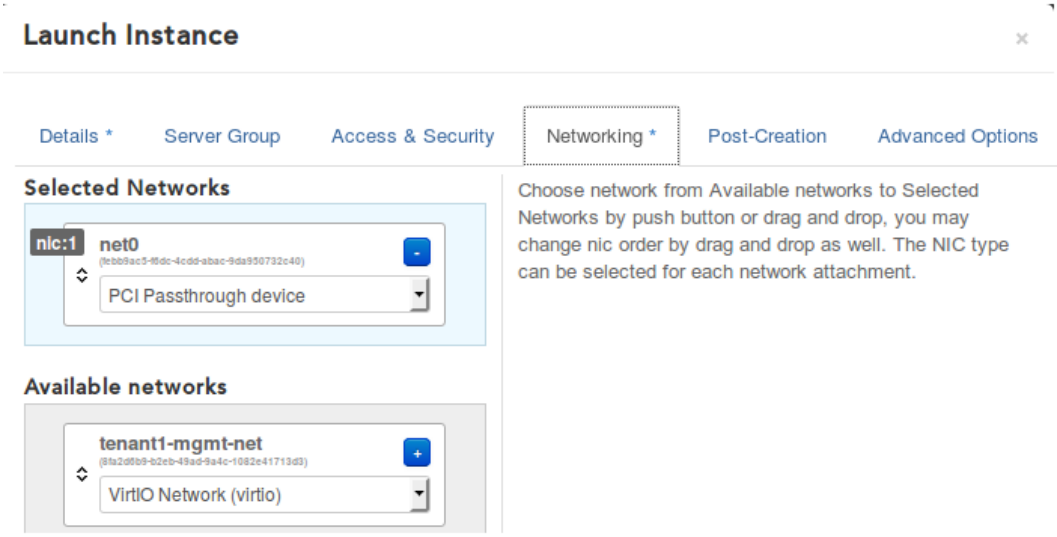
6. Unlock the compute node.

7. Launch the virtual machine.

   a) Log in as user **user1** to the web management interface.

   b) Configure the network interfaces for the new virtual machine.

      Open the **Launch Instance** dialog by clicking the **Launch Interface** button on the **Instances** window. Configure the necessary options for the new virtual machine. In particular, click the **Networking** tab and add a PCI passthrough interface on the tenant network **net0**, as illustrated below. Add other network interfaces as needed.



Click the **Launch** button to proceed.

Passthrough interfaces can be attached from the CLI when booting a new virtual machine, as illustrated below:

```
~(keystone_admin)$ nova boot \
--nic net-id=704e9f3b,vif-model=pci-passthrough \
--flavor small --image cgcs-guest my-new-vm
```

Note:

By default, a VM that requires a PCI passthrough interface is instantiated only if a host NUMA node with a directly attached PCI interface is available. You can specify best-effort instantiation by using an extra specification. For more information, see *Specifying Best Effort for PCI NUMA Node Affinity* on page 152.

The new virtual machine instance is up now. It has a PCI passthrough connection to the **net0** tenant network identified with VLAN ID 10.

Access switches must be properly configured to ensure that virtual machines using PCI-passthrough or SR-IOV Ethernet interfaces (the latter discussed in *SR-IOV Ethernet Interfaces* on page 33) have the expected connectivity. In a common scenario, the virtual machine using these interfaces connects to external end points only, that is, it does not connect to other virtual machines in the same cluster. In this case:

- Traffic between the virtual machine and the access switch can be tagged or untagged.
- The connecting port on the access switch is part of a port-based VLAN.
- If the port is tagged, the allowed VLAN ID range must not overlap with VLAN ID ranges used by the AVS ports.
- The port-based VLAN provides the required connectivity to external switching and routing equipment needed by guest applications to establish connections to the intended end points.

For connectivity to other virtual machines in the cluster the following configuration is also required:

- The VLAN ID used for the tenant network, 10 in this example, and the default port VLAN ID of the access port on the switch are the same. This ensures that incoming traffic from the virtual machine is tagged internally by the switch as belonging to VLAN ID 10, and switched to the appropriate exit ports.
- The target virtual machines are reachable through another port on the compute node, which is managed by the AVS.
- That other port is configured as usual, as a VLAN trunk port, and the tenant network's VLAN ID (10) is included in the tagged range. This ensures that VLAN 10 is common to both the passthrough/SR-IOV interface and the AVS port.

SR-IOV Ethernet Interfaces

A SR-IOV Ethernet interface is a physical PCI Ethernet NIC that implements hardware-based virtualization mechanisms to expose multiple virtual network interfaces that can be used by one or more virtual machines simultaneously.

The PCI-SIG Single Root I/O Virtualization and Sharing (SR-IOV) specification defines a standardized mechanism to create individual virtual Ethernet devices from a single physical Ethernet interface. For each exposed virtual Ethernet device, formally referred to as a *Virtual Function* (VF), the SR-IOV interface provides separate management memory space, work queues, interrupts resources, and DMA streams, while utilizing common resources behind the host interface. Each VF therefore has direct access to the hardware and can be considered to be an independent Ethernet interface.

When compared with a PCI Passthtrough Ethernet interface, a SR-IOV Ethernet interface:

- Provides benefits similar to those of a PCI Passthtrough Ethernet interface, including lower latency packet processing.
- Scales up more easily in a virtualized environment by providing multiple VFs that can be attached to multiple virtual machine interfaces.
- Shares the same limitations, including the lack of support for LAG, QoS, ACL, and live migration.
- Has the same requirements regarding the VLAN configuration of the access switches.
- Provides a similar configuration workflow when used on HP Helion OpenStack Carrier Grade.

It is suggested that you read *PCI Passthrough Ethernet Interfaces* on page 29 first. Most configuration steps are similar, with the difference that you use **pci-sriov** instead of **pci-passthrough** when defining the network type of an interface.

To review the current PCI passthrough and SR-IOV device assignments for a given provider network, see *Displaying Provider Network Information* on page 42.

In the following example a new virtual machine is launched by user **user1** on tenant **tenant1**, with a VF interface connected to the tenant network **tenant1-net1** identified with VLAN ID 20.

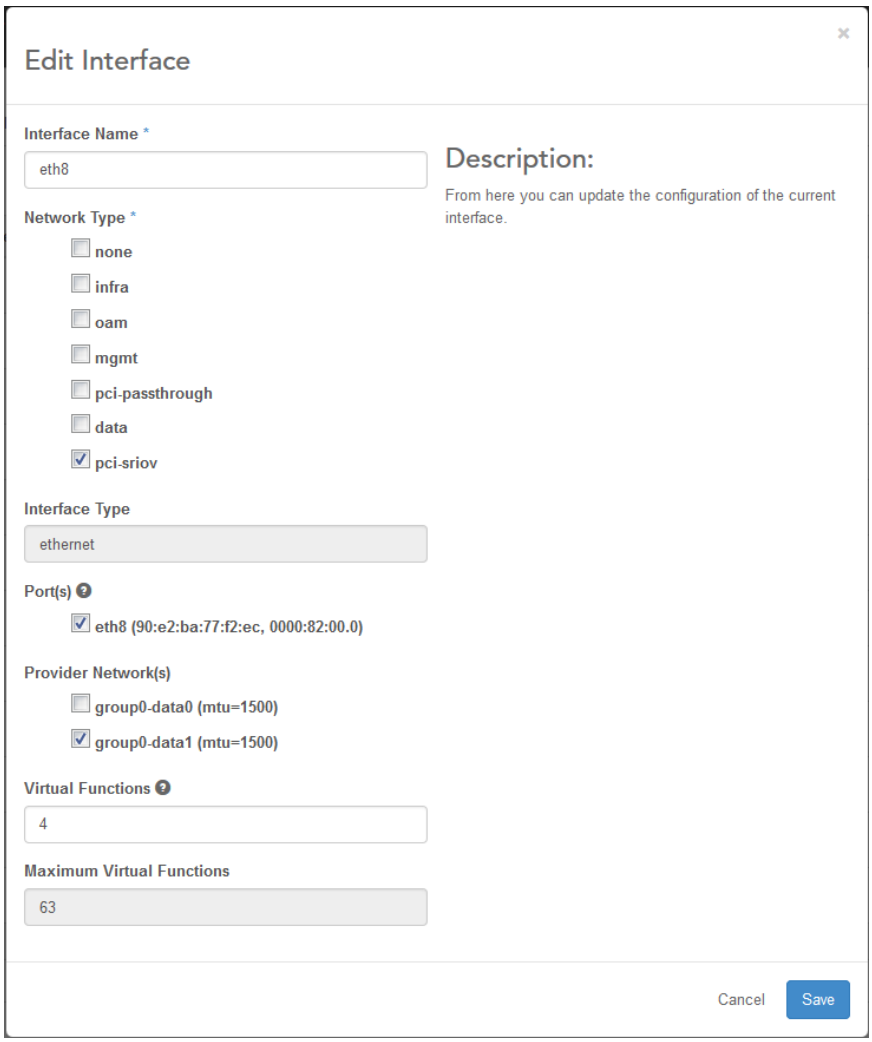
Note:

To use PCI passthrough or SR-IOV interfaces, you must have Intel VT-d features enabled in the BIOS.

The exercise assumes that the underlying provider network **group0-data0** exists already, and that VLAN ID 20 is a valid segmentation ID assigned to **tenant1**.

1. Log in as the **admin** user to the web management interface.
2. Lock the compute node you want to configure.
3. Configure the Ethernet interface to be used as a SR-IOV interface.

   Select the **System Inventory** window from the **Admin** panel, click the **Hosts** tab, click the name of the compute node where the PCI interface is available, click the **Interfaces** tab, and finally click the **Edit Interface** button associated with the interface you want to configure. Fill in the window as illustrated below:



In this example, Ethernet interface **eth8** is assigned the network type *pci-sriov*, enabled to use 4 VFs, and configured as connected to provider network **group0-data0**. Click the **Save** button to proceed.

The Maximum number of VFs displayed in this form is a read-only item which is auto-discovered by the system by inspecting the specific NIC model. This value is reported as 0 when the selected interface does not support SR-IOV.

The interface can also be configured from the CLI as illustrated below:

```
~(keystone_admin)$ system host-if-modify \
-nt pci-sriov -N 4 -p group0-data0 compute-0 eth8
```

Note:

By default, a VM that requires an SR-IOV interface is instantiated only if a host NUMA node with a directly attached PCI interface is available. You can specify best-effort instantiation by using an extra specification. For more information, see *Specifying Best Effort for PCI NUMA Node Affinity* on page 152.

4. Create the **tenant1-net1** tenant network.

   You must ensure that:

   - The **tenant1** tenant has access to the tenant network, either assigning it as the owner, or by enabling the tenant network's shared flag.
   - The segmentation ID is set to 20.

5. Configure the access switch.

   Configure the physical port on the access switch used to connect to Ethernet interface **eth8** as an access port with default VLAN ID of 20.

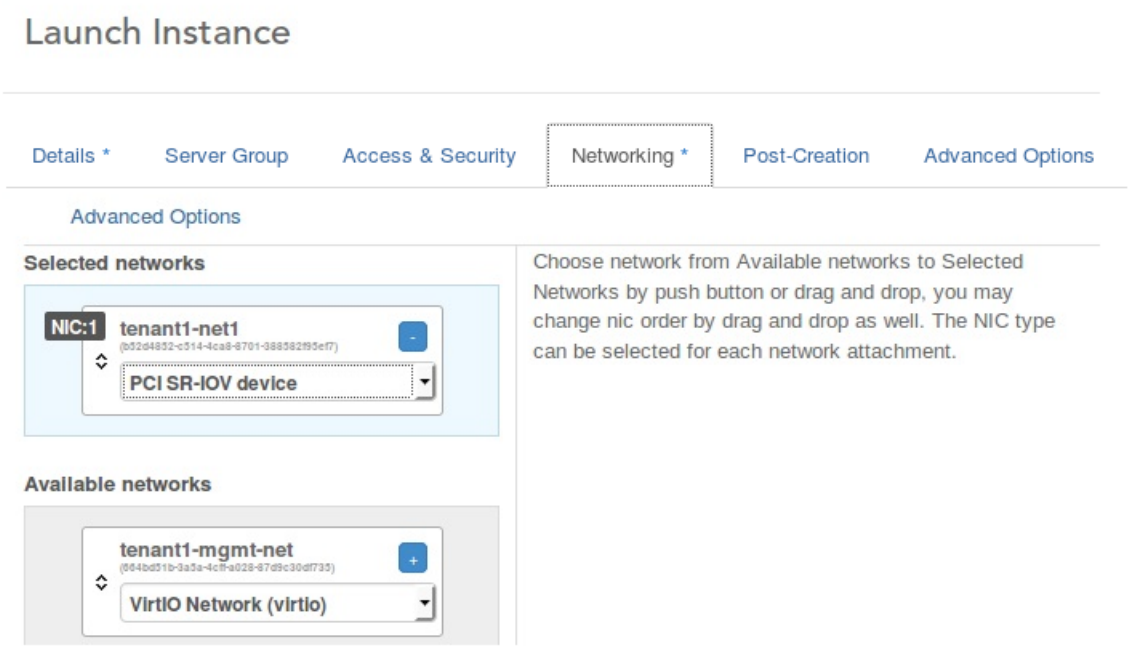
6. Unlock the compute node.

7. Launch the virtual machine.

   a) Log in as user **user1** to the web management interface.

   b) Configure the network interfaces for the new virtual machine.

   Open the **Launch Instance** dialog by clicking the **Launch Interface** button on the **Instances** window. Configure the necessary options for the new virtual machine. In particular, click the **Networking** tab and add a SR-IOV interface on the tenant network **tenant1-net1**, as illustrated below. Add other network interfaces as needed, there are as many SR-IOV ports available as VFs have been enabled.



   Click the **Launch** button to proceed.

   SR-IOV interfaces can be attached from the CLI when booting a new virtual machine, as illustrated below:

```
~(keystone_admin)$ nova boot \
--nic net-id=704e9f3b,vif-model=pci-sriov --flavor small \
--image cgcs-guest my-new-vm
```

The new virtual machine instance is up now. It has a SR-IOV VF connection to the **tenant1-net1** tenant network identified with VLAN ID 20.

Access switches must be properly configured to ensure that virtual machines using PCI-passthrough or SR-IOV Ethernet interfaces (the latter discussed in *SR-IOV Ethernet Interfaces* on page 33) have the expected connectivity. In a common scenario, the virtual machine using these interfaces connects to external end points only, that is, it does not connect to other virtual machines in the same cluster. In this case:

- Traffic between the virtual machine and the access switch can be tagged or untagged.
- The connecting port on the access switch is part of a port-based VLAN.
- If the port is tagged, the allowed VLAN ID range must not overlap with VLAN ID ranges used by the AVS ports.
- The port-based VLAN provides the required connectivity to external switching and routing equipment needed by guest applications to establish connections to the intended end points.

For connectivity to other virtual machines in the cluster the following configuration is also required:

- The VLAN ID used for the tenant network, 10 in this example, and the default port VLAN ID of the access port on the switch are the same. This ensures that incoming traffic from the virtual machine is tagged internally by the switch as belonging to VLAN ID 10, and switched to the appropriate exit ports.
- The target virtual machines are reachable through another port on the compute node, which is managed by the AVS.
- That other port is configured as usual, as a VLAN trunk port, and the tenant network's VLAN ID (10) is included in the tagged range. This ensures that VLAN 10 is common to both the passthrough/SR-IOV interface and the AVS port.

Network Planning

When planning the deployment of HP Helion OpenStack Carrier Grade, it is important to understand the available networking options and how to take advantage of them for the benefit of the end users. The sections that follow provide useful networking guidelines.

The Internal Management Network

The internal management network is a private network, visible only to the hosts in the cluster.

You must consider the following guidelines:

- By default, the management network is used for PXE booting of new hosts, and therefore must be untagged. If, for deployment reasons, the management network needs to be on a VLAN-tagged network, you can configure the optional untagged PXE boot network for PXE booting of new hosts.
- You can use any 1G or 10 G interface on the hosts to connect to this network, provided that the interface supports network booting and can be configured from the BIOS as the primary boot device.
- You can choose dynamic or static IP address assignment. If you use static assignment, you must manually assign IP addresses on the management network for the other hosts in the cluster using the `system host-add` or `system host-update` commands.

  Note:

  The infrastructure network is automatically assigned the same type of address assignment as the management network.
- For the IPv4 address plan, use a private IPv4 subnet as specified in RFC 1918. This helps prevent unwanted cross-network traffic on this network.

  It is suggested that you use the default subnet and addresses provided by the controller configuration script.
- You can assign a range of addresses on the management subnet for use by HP Helion OpenStack Carrier Grade. If you do not assign a range, HP Helion OpenStack Carrier Grade takes ownership of all available addresses.
- The HP Helion OpenStack Carrier Grade controllers use IP multicast messaging on the management, infrastructure, and OAM networks. To prevent loss of controller synchronization, ensure that the switches and other devices on these networks are configured with appropriate settings.

Changing a Management Interface to Aggregated

When compute nodes are provisioned, the Ethernet interface the node booted over is automatically assigned the management network type.

To configure a management LAG interface you first need to remove the management network type from the existing management Ethernet interface and then add a new AE interface, specifying the mgmt network type, ae interface type, 802.3 AE mode, transmit hash policy and the slave interfaces.

The node must be locked to edit an interface.

From the command line, you would first delete and then recreate the management interface.

```
~(keystone_admin)$ system host-if-modify â€“nt none node interface
```

where

node

is the name of the node from which to delete an interface.

interface

is the Ethernet interface to delete.

You would then create the new interface.

```
~(keystone_admin)$ system host-if-add -nt mgmt -a 802.3ad -x
 layer2 node interface ae none ports
```

where

node

is the name of the node

interface

is the name to be assigned to the interface

ports

are the Ethernet ports to assign

For example:

```
~(keystone_admin)$ system host-if-add -nt mgmt -a 802.3ad -x layer2
 compute-0 bond0 ae none eth0 eth1
```

The Infrastructure Network

The infrastructure network is a private network visible only to the hosts in the cluster. It is optional unless storage nodes are part of the cluster.

For the most part, the infrastructure network shares the design considerations applicable to the internal management network. It can be implemented using either a 1 Gb or 10 Gb Ethernet interface. It can be VLAN-tagged, enabling it to share an interface with the management or OAM network. It can own the entire IP address range on the subnet, or a specified range.

The infrastructure network supports dynamic or static IP address assignment. The setting is determined by the setting for the management network. If static assignment is used, you must manually assign IP addresses on the infrastructure network for the other hosts in the cluster, using the `system host-addr-add` command.

The decision whether to implement an infrastructure network depends on the infrastructure traffic levels expected. The following table provides general guidelines on when to implement the infrastructure network, using the number of compute nodes in the HP Helion OpenStack Carrier Grade cluster as an indication of expected traffic levels.

Number of Compute Nodes	Infrastructure Network Implementation
2 to 4	No infrastructure network required. All infrastructure traffic is carried over the internal management network.
5 to 8	Use 1 Gb Ethernet
9 or more	Use 10 Gb Ethernet

In addition to the number of compute nodes, other factors must be taken into account when deciding the type of infrastructure network to use. For example, live migration of large, memory-intensive, guest applications is likely

to require additional network resources that might not be available in the current configuration. It is always safe to implement the fastest infrastructure network possible.

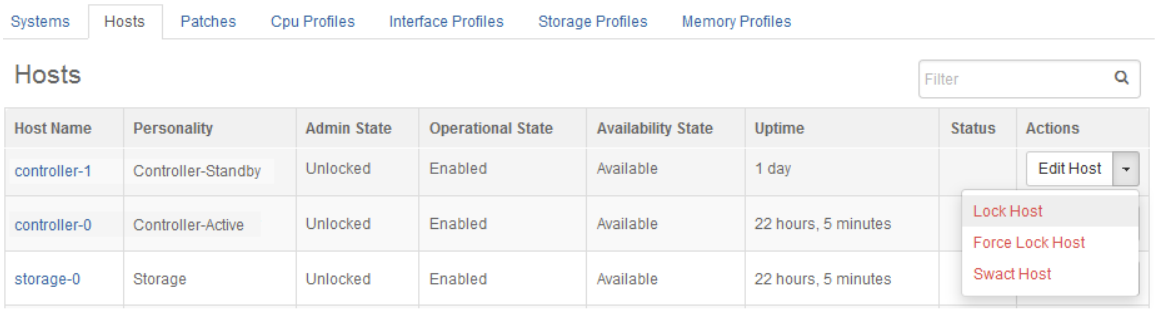
Note:

The HP Helion OpenStack Carrier Grade controllers use IP multicast messaging on the management, infrastructure, and OAM networks. To prevent loss of controller synchronization, ensure that the switches and other devices on these networks are configured with appropriate settings.

Changing the MTU on a small-footprint System

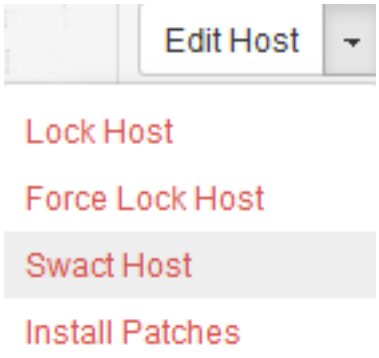
The following steps illustrate how to change the MTU for an interface for an infrastructure network.

1. Lock Controller-1.
   a) From **System** > **Instances**, select the **Hosts** tab.
   b) From the **Edit** menu for Controller-1, select **Lock Host**.



2. Modify the MTU of an interface on controller-1.
   a) Click on the name of controller-1, then switch to the **Interfaces** tab and select **Edit** for the interface you want to change.
   b) In the **Edit Interface** dialog, change the **MTU** field, then click **Save**.
3. Unlock Controller-1.

   From the **Edit** menu for Controller-1, select **Unlock Host**.
4. Swact host to Controller-1



5. Lock controller-0
6. Modify the MTU of an interface on controller-0
7. Unlock Controller-0

Changing the MTU on a small-footprint System using the CLI

The following steps illustrate how to change the MTU for an interface for an infrastructure network. This example uses eth6 as the interface to be changed.

1. Lock controller-1.

```
~(keystone_admin)$ system host-lock controller-1
```

2. Modify the MTU of an interface on controller-1.

```
~(keystone_admin)$ system host-if-modify controller-1 eth6 --imtu 1600
```

3. Unlock controller-1.

```
~(keystone_admin)$ system host-unlock controller-1
```

4. Swact the host to controller-1.

```
~(keystone_admin)$ system host-swact controller-1
```

5. Lock controller-0.

```
~(keystone_admin)$ system host-lock controller-0
```

6. Modify the MTU of the interface on controller-0.

```
~(keystone_admin)$ system host-if-modify controller-0 eth6 --imtu 1600
```

7. Unlock controller-0.

```
~(keystone_admin)$ system host-unlock controller-0
```

The Board Management Network

The board management network is an optional network used for hardware management facilities.

This network provides access to iLO3, iLO4 or Quanta Integrated Lights Out board management modules optionally installed in the hosts. These modules provide support for remote reset and power control of the hosts, and in some cases for hardware sensor monitoring and reporting.

The board management network can be implemented so that it is accessible externally using the OAM network, or only accessible internally to the hosts in the cluster. This choice is made at system installation, during the configuration controller script. For details, see the *HP Helion OpenStack Carrier Grade Software Installation Guide: The Controller Configuration Script*.

Configuration for External Access

For external access, do not configure a board management network during the controller configuration script (press `n` when prompted).

After completing the script, you must also do the following:

- Configure each iLO module to use static IP addressing.
- Peform additional host and module provisioning, as described in *Host and iLO Module Provisioning for a Board Management Network* on page 40.
- Ensure that the OAM Default Gateway specified during controller configuration has access to the board management network.

Configuration for Internal Access

For internal access, configure a board management network during the controller configuration script (press `y` when prompted, and then specify a VLAN ID and a subnet). This configures a VLAN ID on the controller management interface for use with a board management network. To complete the VLAN, you must connect the modules to VLAN-tagged ports on an internal L2 switch. The modules are assigned IP addresses from the specified subnet.

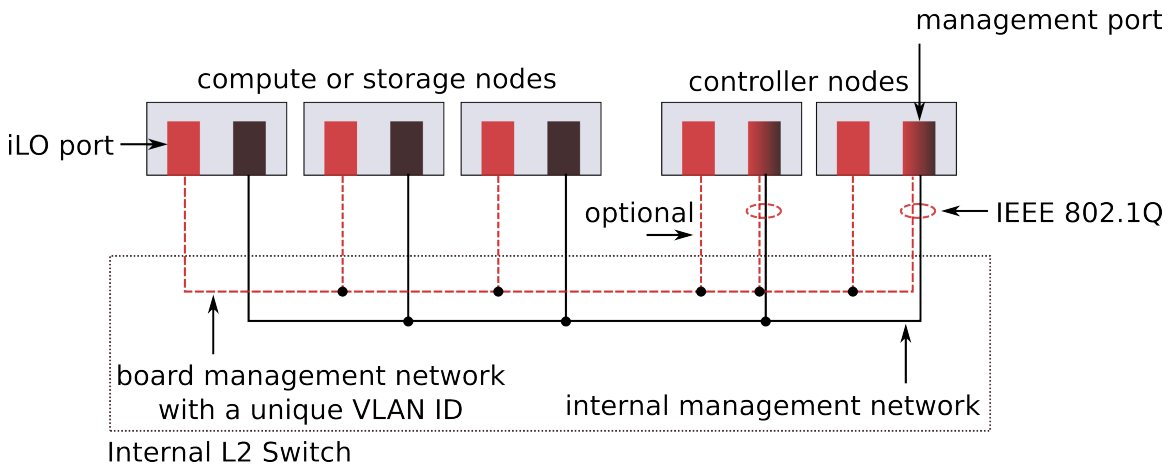


Figure 1: Internal-access board management network

The figure above illustrates how the board management network operates when configured for internal access. Compute nodes connect using iLO modules, while controller nodes use the management port in tagged mode. As indicated in the figure, the controller nodes can optionally attach using their own iLO modules, if available.

- On the HP Helion OpenStack Carrier Grade networks:
  - Designate a VLAN ID for use by the board management network.
  - Configure this VLAN ID as part of the software installation process.
  - Ensure that the internal management network is operational and that all participant hosts are attached to it.
- On the HP Helion OpenStack Carrier Grade internal L2 switch:
  - Configure a VLAN using the designated VLAN ID for the board management network.
  - Add each of the ports used to connect an iLO module to the VLAN.
  - Add the ports used to connect the controllers to the internal management network to the VLAN. You must ensure that these ports are tagged ports for this VLAN, that is, that outgoing board management traffic (toward the controllers) uses IEEE 802.1Q tagging. Other management traffic on this port is untagged.
- Configure each iLO module to use DHCP.
- Peform additional host and module provisioning, as described in *Host and iLO Module Provisioning for a Board Management Network* on page 40.

For details on configuring or provisioning switch ports or iLO modules, consult the user documentation supplied with the equipment.

Host and iLO Module Provisioning for a Board Management Network

For internal or external access, you must perform the following additional provisioning:

- Provision each iLO module with a username and password for secure access.
- Provision each host with the MAC address of the attached iLO module. This is required so that the system can associate the module's assigned IP address with the correct host.
- For external-access configurations only, provision each host with the IP address of the attached iLO module.
- Provision each host with the username and password of the attached iLO module. This is required for controller access to the module.

For module provisioning, consult the user documentation supplied with the equipment.

For host provisioning, you can use the **Board Management** tab on the **Edit Host** dialog box, or the CLI. For more information, see the *HP Helion OpenStack Carrier Grade Software Installation Guide*.

The OAM Network

The OAM (operations, administration, and management) network enables the web administration interface, the command-line management clients, SNMP interface, and the REST APIs to manage the HP Helion OpenStack Carrier Grade cluster.

On systems where an optional board management network is configured for external access, the OAM network also provides access to the board management modules.

You must consider the following guidelines:

- Deploy proper firewall mechanisms to access this network. Ensuring that access to the HP Helion OpenStack Carrier Grade management interfaces is not compromised should be of primary concern.

  HP Helion OpenStack Carrier Grade includes a default firewall for the OAM network, using the **Netfilter** framework. You can optionally configure the system to support additional rules. For more information, see the *HP Helion OpenStack Carrier Grade Installation Guide: Firewall Options*.
- Consider whether the OAM management network needs access to the open Internet. Limiting access to an internal network might be advisable, keeping in mind that access to configured DNS and NTP servers may still be needed.
- VLAN tagging is supported, enabling this network to share an interface with the management or infrastructure networks.
- For the IPv4 address plan, consider the following:
  - The OAM IP floating address is the only address that needs to be visible externally. Therefore you must plan for valid definitions of its IPv4 subnet and default gateway.
  - The physical IPv4 addresses for the controllers don't need to be visible externally, unless you plan to use them during **ssh** sessions to prevent potential service breaks during the connection. You still need to plan for their IPv4 subnet, but you can limit access to them as required.
- The HP Helion OpenStack Carrier Grade controllers use IP multicast messaging on the management, infrastructure, and OAM networks. To prevent loss of controller synchronization, ensure that the switches and other devices on these networks are configured with appropriate settings.

Provider Networks

Provider networks are the payload-carrying networks used implicitly by end users when they move traffic over their tenant networks.

You can review details for existing provider networks using the web administration interface or the CLI. For more information, see *Displaying Provider Network Information* on page 42.

When planning provider networks, you must consider the following guidelines:

- From the point of view of the tenants, all networking happens over the tenant networks created by them, or by the **admin** user on their behalf. Tenants are not necessarily aware of the available provider networks. In fact, they cannot create tenant networks over provider networks not already accessible to them. For this reason, the system administrator must ensure that proper communication mechanisms are in place for tenants to request access to specific provider networks when required.

  For example, a tenant may be interested in creating a new tenant network with access to a specific network access device in the data center, such as an access point for a wireless transport. In this case, the system administrator must create a new tenant network on behalf of the tenant, using a VLAN ID in the provider network's segmentation range that provides connectivity to the said network access point.
- Consider how different offerings of bandwidth, throughput commitments, and class-of-service, can be used by your users. Having different provider network offerings available to your tenants enables end users to diversify their own portfolio of services. This in turn gives the administration an opportunity to put different revenue models in place.
- For the IPv4 address plan, consider the following:
  - Tenant networks attached to a public network, such as the Internet, have to have external addresses assigned to them. Therefore you must plan for valid definitions of their IPv4 subnets and default gateways.

    - As with the OAM network, you must ensure that suitable firewall services are in place on any tenant network with a public address.
- Segmentation ranges defined on a provider network may be owned by the administrator, a specific tenant, or may be shared by all tenants. With this ownership model:
    - A base deployment scenario has each compute node using a single data interface defined over a single provider network. In this scenario, all required tenant networks can be instantiated making use of the available VLANs or VNIs in each corresponding segmentation range. You may need more than one provider network when the underlying physical networks demand different MTU sizes, or when boundaries between provider networks are dictated by policy or other non-technical considerations.
    - Segmentation ranges can be reserved and assigned on-demand without having to lock and unlock the compute nodes. This facilitates day-to-day operations which can be performed without any disruption to the running environment.
- In some circumstances, provider networks can be configured to support VLAN Transparent mode on tenant networks. In this mode VLAN tagged packets are encapsulated within a provider network segment without removing or modifying the guest VLAN tag. For more information, see *VLAN Transparent* on page 44.

Displaying Provider Network Information

You can display details for a provider network from the CLI or the web administration interface.

To view information for a provider network from the CLI, use the following command:

```
~(keystone_admin)$ neutron providernet-show providernet
```

where *providernet* is the name or UUID of the provider network.

For example:

```
~(keystone_admin)$ neutron providernet-show a507bb89-8e78-48e4-8c8a-
d8bac14fc097
```

Viewing PCI Resource Usage from the CLI

To view information about PCI interface resources for a provider network from the CLI, use the following command:

```
~(keystone_admin)$ nova providernet-show providernet
```

where *providernet* is the UUID of the provider network.

For example:

```
~(keystone_admin)$ nova providernet-show a507bb89-8e78-48e4-8c8a-
d8bac14fc097
+--------------------+--------------------------------------+
| Property           | Value                                |
+--------------------+--------------------------------------+
| id                 | a507bb89-8e78-48e4-8c8a-d8bac14fc097 |
| name               | group0-ext0                          |
| pci_pfs_configured | 1                                    |
| pci_pfs_used       | 1                                    |
| pci_vfs_configured | 0                                    |
| pci_vfs_used       | 0                                    |
+--------------------+--------------------------------------+
```

where

pci_pfs_configured

is the number of PCI passthrough interfaces attached to the provider network

pci_pfs_used

is the number of PCI passthrough interfaces allocated to instances

pci_vfs_configured

is the number of SR-IOV interfaces attached to the provider network

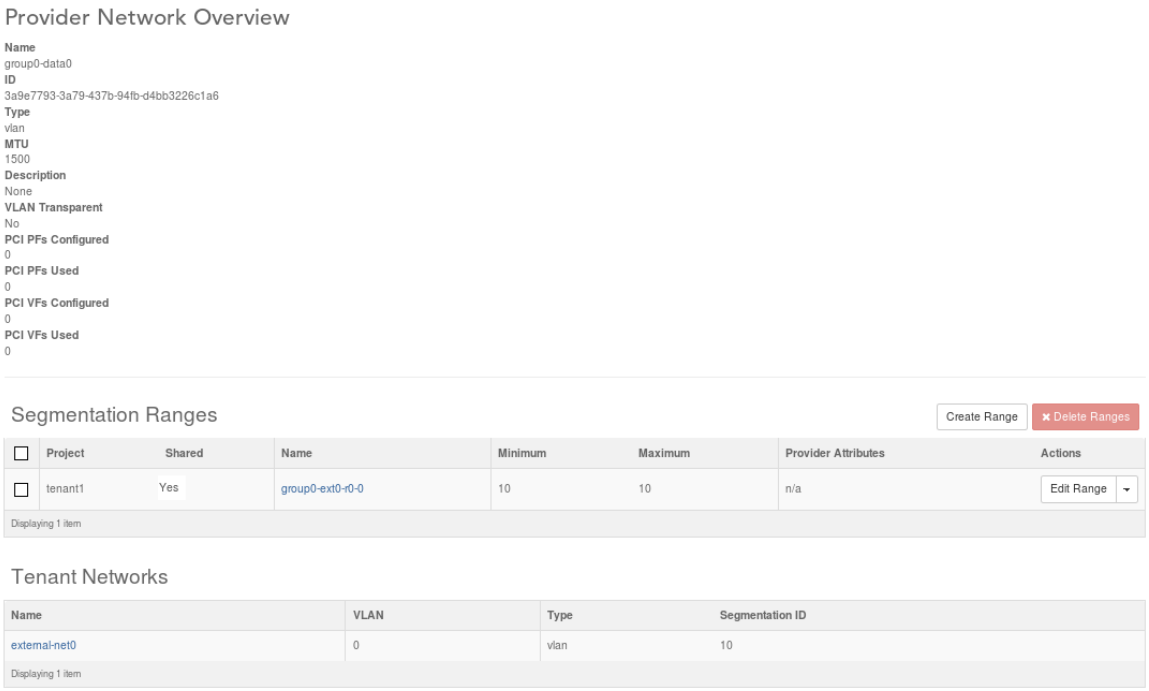
pci_vfs_used

is the number of SR-IOV interfaces allocated to instances

To view the information from the web administration interface, follow these steps:

1. From the left pane menu select **Admin** > **System** > **Networks** to display the **Networks** page.
2. Select the **Provider Networks** tab.
3. Click the name of the provider network to open the **Provider Network Overview** page.

   Details for the provider network are displayed, including any associated segmentation ranges and tenant networks.



Tenant Networks

Tenant networks are logical networking entities visible to tenant users, and around which working network topologies are built.

Tenant networks need support from the physical layers to work as intended. This means that the access L2 switches, providers' networks, and data interface definitions on the compute nodes, must all be properly configured. In particular, when using provider networks of the VLAN or VXLAN type, getting the proper configuration in place requires additional planning.

For provider networks of the VLAN type, consider the following guidelines:

- All ports on the access L2 switches must be statically configured to support all the VLANs defined on the provider networks they provide access to. The dynamic nature of the cloud might force the set of VLANs in use by a particular L2 switch to change at any moment.
- The set of VLANs used by each compute node is not fixed; it changes over time. The current VLAN set in use is determined by the configuration details of the tenant networks, and the scheduling on the compute nodes of the virtual machines that use them. This information is provided to the Neutron's AVS plugin, which then uses it to configure the AVS as required.

When a tenant creates a new network, the Neutron segmentation allocation strategy is to look first for an available segmentation identifier owned by the tenant. If none is available, the search continues over the available shared segmentation identifiers. The allocation process returns an error if no segmentation identifiers are available.

The VLAN ID assigned to a tenant network is fixed for as long as the tenant network is defined in the system. If for some reason the VLAN ID has to change, the tenant network must be deleted and recreated again.

- Configuring a tenant network to have access to external networks (not just providing local networking) requires the following elements:
  - A physical router, and the provider network's access L2 switch, must be part of the same Layer-2 network. Because this Layer 2 network uses a unique VLAN ID, this means also that the router's port used in the connection must be statically configured to support the corresponding VLAN ID.
  - The router must be configured to be part of the same IP subnet that the tenant network is intending to use.
  - When configuring the IP subnet, the tenant must use the router's port IP address as its external gateway.
  - The tenant network must have the **external** flag set. Only the **admin** user can set this flag when the tenant network is created.

For provider networks of the VXLAN type, consider the following guidelines:

- Layer 3 routers used to interconnect compute nodes must be multicast-enabled, as required by the VXLAN protocol.
- To minimize flooding of multicast packets, IGMP and MLD snooping is recommended on all Layer 2 switches. The AVS switch supports IGMP V1, V2 and V3, and MLD V1 and V2.
- To support IGMP and MDL snooping, Layer 3 routers must be configured for IGMP and MDL querying.
- To accommodate VXLAN encapsulation, the MTU values for Layer 2 switches and compute node data interfaces must allow for additional headers. For more information, see *The Ethernet MTU* on page 26.
- To participate in a VXLAN network, the data interfaces on the compute nodes must be configured with IP addresses, and with route table entries for the destination subnets or the local gateway. For more information, see *Configuring Endpoint IP Addresses Using the CLI* on page 77, and *Adding and Maintaining Routes for a VXLAN Network* on page 78.

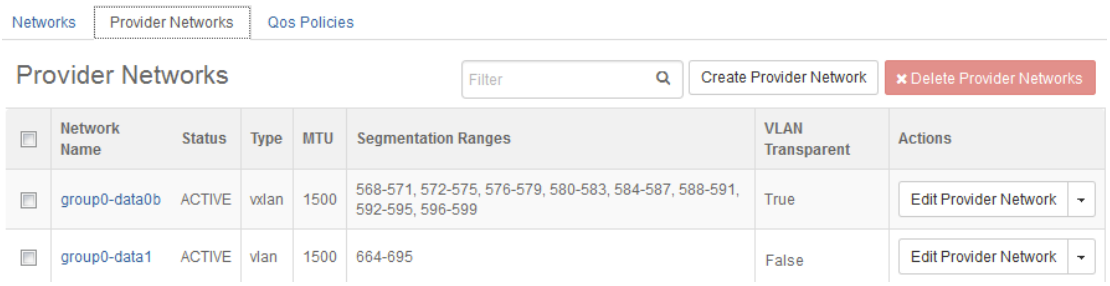
In some circumstances, tenant networks can be configured to use VLAN Transparent mode, in which VLAN tagged packets from the guest are encapsulated within a provider network segment (VLAN) without removing or modifying the guest VLAN tag. For more information, see *VLAN Transparent* on page 44. Alternately, guest VLAN-tagged traffic can be supported by HP Helion OpenStack Carrier Grade tenants explicitly defining one or more VLAN-tagged IP subnets on a single tenant network. With this approach, the guest VLAN-tagged IP subnets have access to all of the services of the virtualized network infrastructure, such as DHCP, virtual routing, meta-data server, etc. For more information, see *HP Helion OpenStack Carrier Grade Overview* on page 6.

VLAN Transparent

A *vlan transparent* tenant network is one that allows VLAN tagged packets to be encapsulated within a provider network segment without removing or modifying the guest VLAN tag. VLAN Transparent is provided in addition to VLAN-tagged Neutron subnets for guest VLANs.

VLAN Transparent must be supported on a provider network before a VLAN Transparent tenant network can be created on that provider network. The **VLAN Transparent** column in the list of defined provider networks on the **Provider Networks** tab of the **System** > **Networks** page indicates availability.

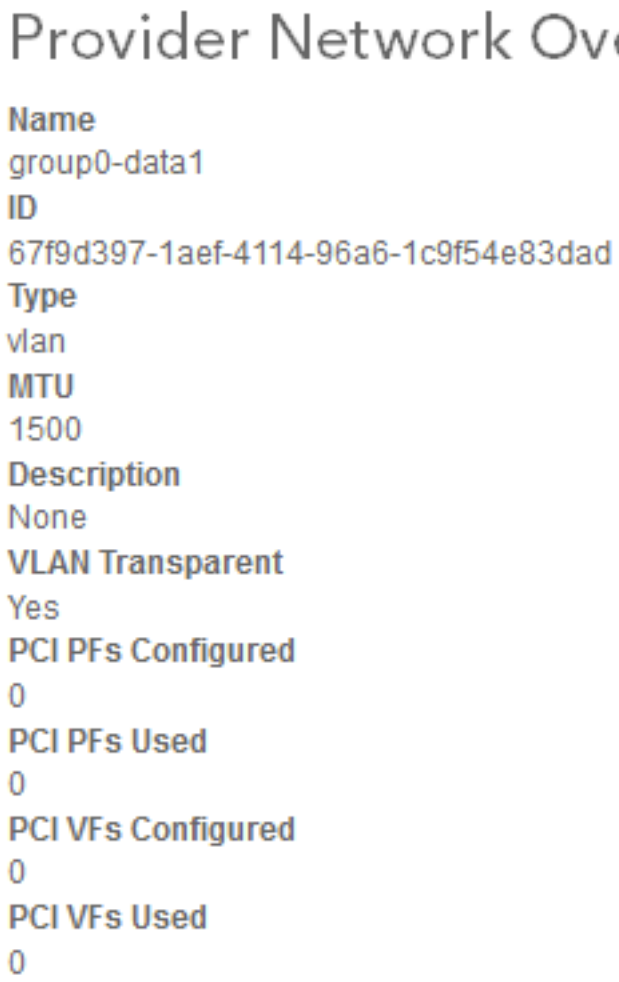
VLAN Priority attributes are propagated to the provider VLAN tag and then restored to the guest VLAN tag.



This information is also available from the following command.

```
~(keystone_admin)$ neutron providernet-show provider1-vt
+------------------+--------------------------------------------------------
+
| Field            | Value
 |
+------------------+--------------------------------------------------------
+
| description      |
 |
| id               | 72a09e68-7170-493e-b9c6-d2bcb881f458
 |
| mtu              | 1500
 |
| name             | provider1-vt
 |
| ranges           | {
  |
|                  |      "name": "provider1-vt-r1-0",
 |
|                  |      "tenant_id": "c17a85ac5283496da1152ce68b79e606",
 |
|                  |      "maximum": 50,
 |
|                  |      "minimum": 59,
 |
|                  |      "shared": false,
 |
|                  |      "id": "8b1fa184-80e0-4555-bd9f-bcf444384189",
 |
|                  |      "description": null
 |
|                  | }
 |
| status           | DOWN
 |
| type             | vlan
 |
| vlan_transparent | True
 |
+------------------+--------------------------------------------------------
+
```

The summary information available by selecting a tenant network name form the **Provider Networks** tab of the **System** > **Networks** page indicates if a tenant network will request VLAN Transparent services from a provider network.



This information is also available from the following command.

```
~(keystone_admin)$ neutron net-show tenant1-vt
+---------------------------+--------------------------------------+
| Field                     | Value                                |
+---------------------------+--------------------------------------+
| admin_state_up            | True                                 |
| id                        | 011cea1e-422f-49d1-9d80-ba864bab361f |
| mtu                       | 1500                                 |
| name                      | tenant1-vt                           |
| provider:network_type     | vlan                                 |
| provider:physical_network | provider1-vt                         |
| provider:segmentation_id  | 50                                   |
| router:external           | False                                |
| shared                    | False                                |
| status                    | ACTIVE                               |
| subnets                   | 3bff6cb0-8422-4596-857f-a02d68a051b1 |
|                           | dedcd9e0-d4ee-4306-a6bf-194104baa9b4 |
| tenant_id                 | c17a85ac5283496da1152ce68b79e606     |
| vlan_transparent          | True                                 |
+---------------------------+--------------------------------------+
```

For more information on setting up provider networks, see *HP Helion OpenStack Carrier Grade Software Installation Guide: Configuring Provider Networks* and *HP Helion OpenStack Carrier Grade Software Installation Guide: Configuring Provider Networks Using the CLI*. For more information on setting up tenant networks, see *Creating Tenant Networks* on page 47.

Table 2: VLAN Transparent Compatibility

	802.1p	802.1q	802.1ad	QinQ
Flat Provider Network	Yes	Yes	Yes	Yes
VLAN Provider Network	Yes	Yes	Yes	Yes
VXLAN Provider Network	Yes	Yes	Yes[1]	Yes
VLAN Tagged Subnets	Yes	Yes	No	No

Creating Tenant Networks

You can use the CLI or Web interface to set up tenant networks and their associated IP subnets.

To create a tenant network using the CLI, use the following command:

```
~(keystone_admin)$ neutron net-create --tenant-id UUID \
--provider:physical_network=network \
--provider:network_type=type \
--provider:segmentation_id=segment \
--shared --router:external --admin-state-down \
--vlan-transparent=state
 \
name
```

where

UUID

is the UUID of the tenant network

network

is the provider network to use

type

is the type of network to create, this can be one of flat, vlan or vxlan

segment

is the provider network segment ID to use

state

is either "True" or "False", indicating if this tenant network will attempt to use VLAN Transparent mode. For more information, see *VLAN Transparent* on page 44.

name

is the name of the tenant network

To create a tenant subnet using the CLI, use the following command:

```
~(keystone_admin)$ neutron subnet-create --tenant-id UUID \
--name name --gateway gateway --disable-dhcp external-net externalIP
```

[1] VLAN priority attributes not propagated to IP header differentiated services field.

where

UUID

is the ID of the tenant network the subnet is being created for

name

is the name of the subnet

gateway

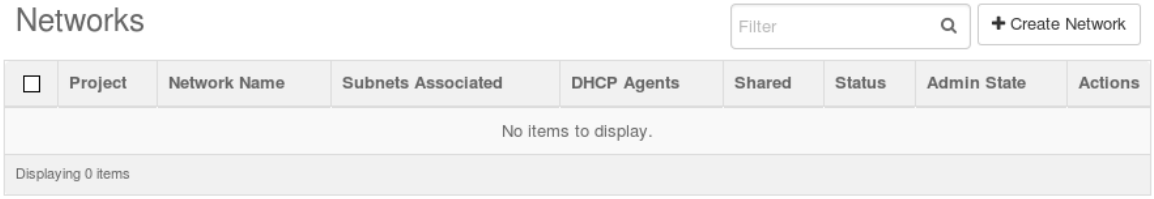
is the IP address of the gateway

address

is the IP address/bitmap of the external network

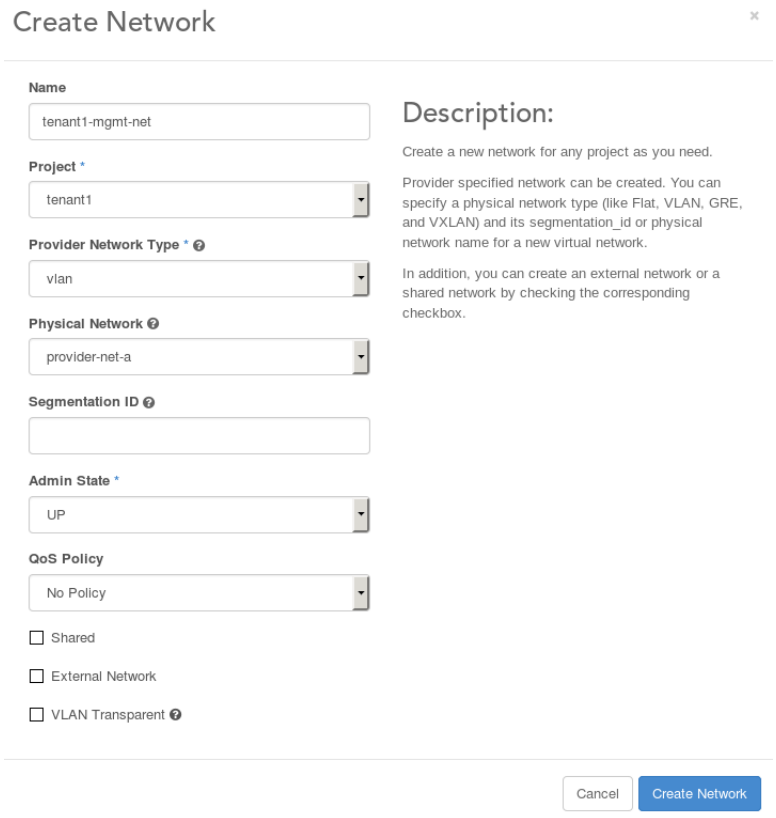
1. List the tenant networks currently defined on the system.

   Select **Networks** in the System Panel section of the **Admin** tab to open the **Networks** page.



2. Create the tenant network.

   From the **Networks** tab, click **Create Network** and fill in the form as illustrated below:



   Click **Create Network** to commit the changes.

3. Create any required subnets.

a) Select **Networks** in the System Panel section of the **Admin** tab to open the **Networks** page.
b) Select the name of the tenant network you just created.
c) Click **Create Subnet**.
d) Complete the forms on the **Subnet** and **Subnet Details** tabs, then click **Create**

Virtual Routers

Virtual routers provide internal and external network connectivity for tenants.

The user associated with a tenant can add a designated number of virtual routers (Neutron routers) to the tenant. The maximum number is specified in the quotas for the tenant.

The virtual router automatically provides a connection to system services required by the tenant, such as the metadata service that supplies instances with user data. You can configure the virtual routers with interfaces to tenant networks to provide internal and external connectivity.

A virtual router can be implemented as a centralized router, or a distributed virtual router (DVR).

Note:

Only the **admin** user can specify a distributed router. For other tenants, this choice is not available, and a centralized router is implemented by default. The **admin** user can change a centralized router to a distributed router on behalf of other tenants (see *Virtual Router Administration* on page 55).

A distributed router cannot be converted to a centralized router.

Centralized

A centralized virtual router is instantiated on a single compute node. All traffic using the router must pass through the compute node.

Distributed

A distributed virtual router is instantiated in a distributed manner across multiple hosts. Distributed virtual routers provide more efficient routing than standard virtual routers for east-west (tenant-to-tenant) or floating IP address traffic. Local traffic is routed within the local host, while external L3 traffic is routed directly between the local host and the gateway router.

To implement the distributed model, a centralized-portion of the router is still deployed on one host. The centralized portion manages north-south (external network) traffic and source network address translation (SNAT) traffic, as well as agents deployed on other hosts. The agents offload the centralized router for east-west (tenant-to-tenant) routing and floating IP network address translation.

You can enable SNAT on a virtual router. For more information, see *Configuring SNAT on a Virtual Router* on page 54

To add a virtual router, see *Adding Virtual Routers* on page 49. To create interfaces, see *Adding Virtual Router Interfaces* on page 51.

Adding Virtual Routers

You can add virtual routers to a tenant using the web administration interface or the CLI.

To add a router to a tenant, you must be logged in as the user associated with the tenant.

As an alternative to the web administration interface, you can use CLI commands to add a virtual router. First, become the appropriate keystone user. The examples in this section assume you are the **admin** user.

To create a router from the CLI, use the `neutron router-create` command. For example:

```
~(keystone_admin)$ neutron router-create router_name --distributed=True
```

where *router_name* is a name assigned to the router.

This example creates a distributed virtual router. If the --distributed option is omitted, the default setting (centralized) is used.

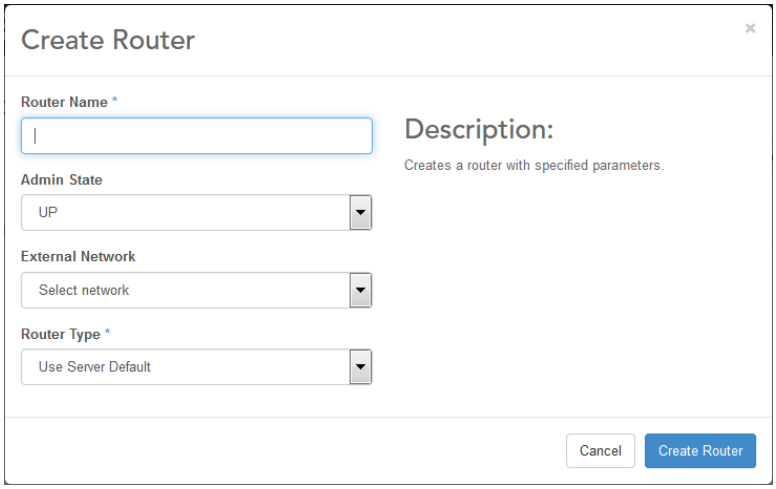
Note:

Only the **admin** tenant can add a distributed virtual router.

1. Log in as the user associated with the tenant.

   The following instructions assume that the **admin** tenant is logged in.
2. Select **Project** > **Network** > **Routers**.
3. On the **Routers** page, click **Create Router**.

   The Create Router dialog box appears.



4. Provide a **Router Name**.
5. Optional: To add a connection to a tenant network configured for external access, select from the **External Network** drop-down menu.

   This adds a gateway for the router. Alternatively, you can add a gateway after the router is created. For more information, see *Adding Virtual Router Interfaces* on page 51.
6. Select the **Router Type**.

   Note:

   The **Router Type** field is shown only for the **admin** user. For other tenants, a centralized router is added. The **admin** user can change this setting on behalf of another tenant after the router has been added (see *Virtual Router Administration* on page 55).

   Use Server Default

   Use the default setting specified on the controller (for a standard HP Helion OpenStack Carrier Grade installation, centralized).

   Centralized

   Add a centralized virtual router.

   Distributed

   Add a distributed virtual router (DVR).

   For more information about these choices, see *Virtual Routers* on page 49.
7. To save your settings and close the dialog box, click **Create Router**.

To attach router interfaces to tenant networks, see *Adding Virtual Router Interfaces* on page 51.

Adding Virtual Router Interfaces

You can create router interfaces to tenant networks using the web administration interface or the CLI.

For more information about virtual routers, see *Virtual Routers* on page 49.

Before you can create interfaces to tenant networks, you must create the tenant networks and associated subnets. For examples, see the *HP Helion OpenStack Carrier Grade Reference Deployment Scenarios*. For more information about tenant networks and subnets, see *Tenant Networks* on page 43 and *Managed and Unmanaged Subnets* on page 55.

As an alternative to the web administration interface, you can use CLI commands to add router interfaces. First, become the appropriate keystone user. The examples in this section assume you are the **admin** user.

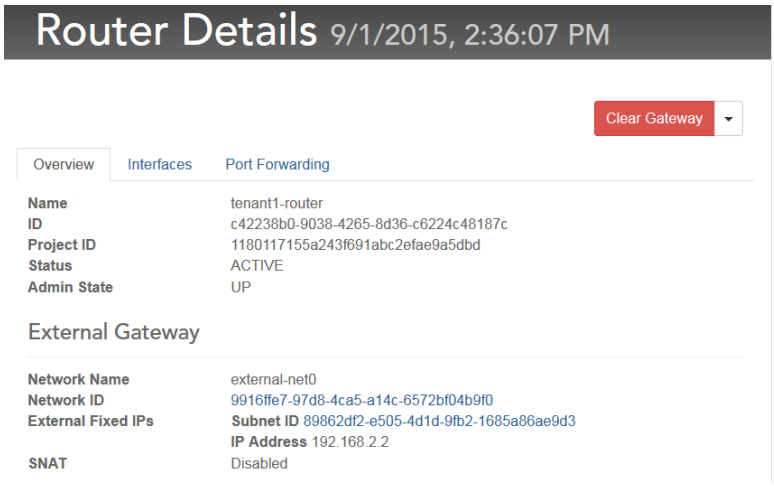
Then use the `neutron router-interface-add` command to add interfaces. For example:

```
~(keystone_admin)$ neutron router-interface-add router_id subnet-id
```

where *router_id* is the name or UUID of the router, and *subnet_id* is the name or UUID of the subnet to which you want to attach an interface. By default, the interface is assigned the gateway address on the subnet.

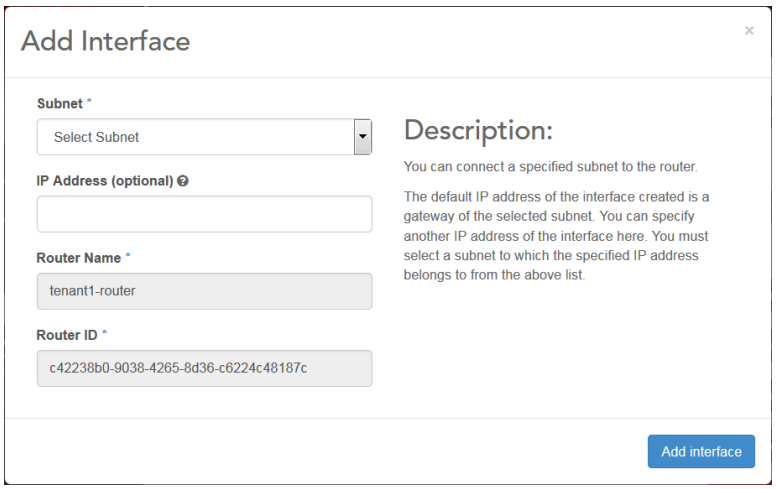
1. Log in as the user associated with the tenant.
2. Select **Project** > **Network** > **Routers**.
3. In the list of existing routers, click the name of the router to be configured.

   The **Router Details** page appears.



4. On the **Interfaces** tab, click **Add Interface**.

   The **Add Interface** dialog box appears.



Subnet

Use the drop-down list to select from existing subnets defined for the tenant. Each subnet is identified by a tenant network name, followed by a subnet address and mask created for the tenant network.

IP Address (optional)

You can optionally specify an IP address for the interface. By default, the gateway address for the subnet is used (for example, 192.168.102.1).

Router Name

This field is shown for information only.

Router ID

This field is shown for information only.

5. To save your settings and close the dialog box, click **Add Interface**.

   The new interface is shown in the **Router Details**.

Adding a Gateway to a Virtual Router

You can configure a gateway for a virtual router as a tenant or an administrator. Users with administrator privileges have access to additional controls.

You can use either the web administration interface or the CLI to configure a gateway. Steps for the web administration interface are given later in this section.

To use the CLI, become the appropriate keystone user. The examples in this section assume you are the **admin** user.

To add a gateway interface, use a command of the following form:

```
~(keystone_admin)$ neutron router-gateway-set router_id externalnet_id \
[--disable-snat] [--fixed-ip gateway_addr]
```

where

router_id

is the name or UUID of the router

externalnet_id

is the name or UUID of a tenant network configured to provide external connections

gateway_addr

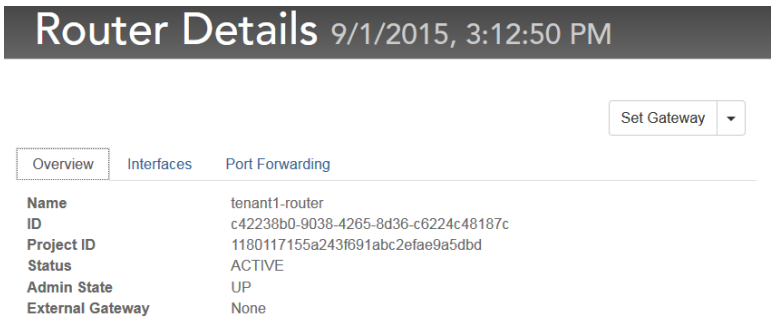
is the gateway interface IP address. If *gateway_addr* is not specified, the server allocates the next available external IP address.

Note:

The fixed-ip and disable-snat options require administrator privileges. For more about the SNAT option, see *Configuring SNAT on a Virtual Router* on page 54.

1. Log in as the user associated with the tenant.
2. Open the **Routers** page.
   - To open the page with tenant privileges, select **Project** > **Network** > **Routers**.
   - To open the page with administrator privileges, select **Admin** > **System** > **Routers**.
3. In the list of existing routers, click the name of the router to be configured.

   The **Router Details** page appears.

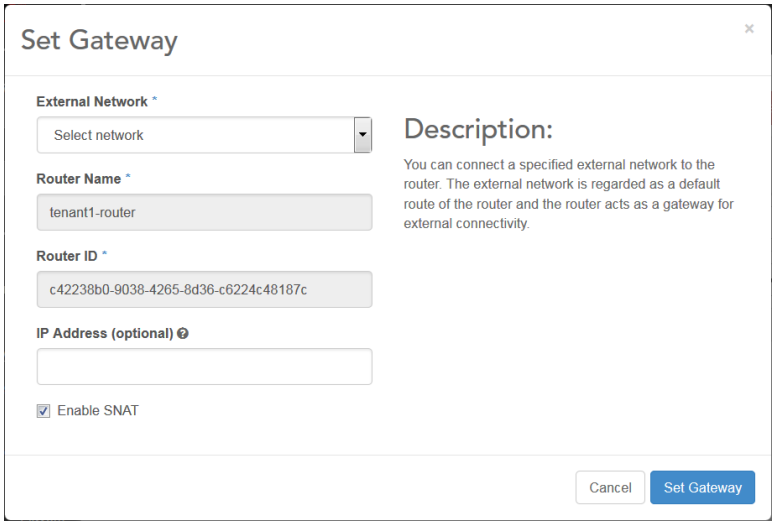


4. On the **Interfaces** tab, click **Set Gateway**.

   Note:

   If a gateway has already been added, a **Clear Gateway** button is present instead.

   The **Set Gateway** dialog box appears.



External Network

Select a tenant network configured to provide external access.

Router Name

This field is shown for information only.

Router ID

This field is shown for information only.

IP Address (optional)

This field is shown only for users with administrator privileges. You can optionally specify an IP address for the gateway interface. By default, the server allocates the next available external IP address.

Enable SNAT

This control is shown only for users with administrator privileges. For more about the SNAT option, see *Configuring SNAT on a Virtual Router* on page 54

To save the gateway gettings and close the dialog box, click **Set Gateway**.

Configuring SNAT on a Virtual Router

Source Network Address Translation (SNAT) provides network address translation for internally initiated UDP and TCP traffic flows. A virtual router with SNAT enabled translates between IP addresses and ports used on an internal network, and an IP address and ports used on an external network.

You can enable or disable Source Network Address Translation for each individual external network connection on a virtual router. By default, SNAT is enabled for all external networks.

When SNAT is enabled, the virtual router maintains network translation tables for all internally-initiated UDP and TCP traffic flows. An instance connected to an internal tenant network can initiate connections to servers on an external, usually public, network. SNAT does *not* allow external servers to initiate connections to external addresses mapped to internal addresses. For this capability, the use of floating IP addresses is required.

System administrators can enable or disable SNAT using the following command:

```
~(keystone_admin)$ neutron router-update routername --external_gateway_info
 type=dict \
network_id=externalnetwork,enable_snat=boolean
```

where:

routername

is the name or UUID of the virtual router

externalnetwork

is the name or UUID of the external network

boolean

is true (to enable SNAT) or false (to disable SNAT)

Caution:

For SNMP implemented on guests, the following limitations apply when SNAT is enabled:

- An SNMP agent on an internal network cannot receive requests from SNMP managers on external networks.
- An SNMP manager on an internal network cannot receive traps from SNMP agents on external networks, even if communication is initiated by the manager. This is because SNMP traps are sent to UDP port 162, not to the source port.

Only agent responses to requests from a manager on the internal network are processed appropriately.

To overcome these limitations, you can use floating IP addresses or make internal addresses public. For more information, see *Connecting Virtual Machines to External Networks* on page 168.

Virtual Router Administration

Users with administrative privileges can review and edit router information for all tenants.

The **Routers** page accessed from **Admin** > **System** > **Routers** provides central control of all virtual routers. From this page, you can edit router names, change router types, change the status of routers (**Up** or **Down**), and delete routers.

Only users with administrative privileges can convert a centralized router to a distributed router.

Note:

A distributed router cannot be converted to a centralized router.

An administrative user can delete routers on any tenant. A non-administrative user can delete routers on the tenant associated with the user.

As an alternative to the web administration interface, you can use the CLI. The following examples assume you have become the **admin** user.

- To list routers:

```
~(keystone_admin)$ neutron router-list
```

- To show details for a router:

```
~(keystone_admin)$ neutron router-show router_id
```

  where *router_id* is the name or UUID of the router.
- To update a router:

```
~(keystone_admin)$ neutron router-update router_id --distributed=True
```

  where *router_id* is the name or UUID of the router. This example updates a router to make it a distributed virtual router.
- To delete a router:

```
~(keystone_admin)$ neutron router-delete router_id
```

Managed and Unmanaged Subnets

Use the *System Managed Subnet* and *Enable DHCP* subnet attributes to determine how IP addresses are allocated and offered on an IP subnet.

With the proper configuration in place, DHCP services can be provided by the built-in Neutron DHCP server, by a standalone server available from an external network infrastructure, or by both.

When creating a new IP subnet for a tenant network you can specify the following attributes:

System Managed Subnet

When this attribute is enabled, the subnet is *system managed*. the Neutron service automatically allocates an IP address from the address allocation pools defined for the subnet to any new virtual machine instance with a virtual Ethernet interface attached to the tenant network. Once allocated, the pair (MAC address, IP address) is registered in the Neutron database as part of the overall registration process for the new virtual machine.

When the system managed subnet attribute is disabled, the subnet is *unmanaged*. No automatic allocation of IP addresses takes place, and the Neutron DHCP service for the subnet is disabled. Allocation of IP addresses for new virtual machines must be done at boot time using the CLI or the API interfaces.

Enable DHCP

When this attribute is enabled, a virtual DHCP server becomes available when the subnet is created. It uses the (MAC address, IP address) pairs registered in the Neutron database to offer IP addresses in response to DHCP discovery requests broadcast on the subnet. DHCP discovery requests from unknown MAC addresses are ignored.

The Neutron DHCP server can only be enabled on system managed subnets. DHCP services for unmanaged subnets, if required, must be provisioned by external, non-Neutron, DHCP servers.

When the DHCP attribute is disabled, all DHCP and DNS services, and all static routes, if any, must be provisioned externally.

Allocation Pools

This a list attribute where each element in the list specifies an IP address range, or address pool, in the subnet address space that can be used for dynamic offering of IP addresses. By default there is a single allocation pool comprised of the entire subnet's IP address space, with the exception of the default gateway's IP address.

An external, non-Neutron, DHCP server can be attached to a system managed subnet to support specific deployment needs as required. For example, it can be configured to offer IP addresses on ranges outside the Neutron allocation pools to service physical devices attached to the tenant network, such as testing equipment and servers.

Allocation pools can only be specified on system managed subnets.

On the web administration interface you set these attributes in the **Subnet Detail** tab of the **Create Subnet** window. On the CLI you set these attributes as illustrated in the following examples.

Create a new system managed subnet, DHCP-enabled, and a custom allocation pool:

```
$ neutron subnet-create --name my-subnet \
--allocation-pool start=172.18.0.100,end=172.18.0.150 \
my-tenant-net 172.18.0.0/24
```

Create a new system managed subnet with DHCP service disabled:

```
$ neutron subnet-create --name my-subnet \
--disable-dhcp \
my-tenant-net 172.18.0.0/24
```

Create a new unmanaged subnet:

```
$ neutron subnet-create --name my-subnet \
--unmanaged --disable-dhcp \
my-tenant-net 172.18.0.0/24
```

Accessing the Metadata Server

Access to the system's metadata server is available using the well known URL http://169.254.169.254. In the HP Helion OpenStack Carrier Grade implementation, access to this address is provided by a virtual router attached to the tenant on which the access request is made. Virtual routers are automatically configured as proxies to the metadata service.

The following requirements must be satisfied in order for a guest application to access the metadata service:

1. There is a route table entry to route traffic destined to the 169.254.169.254 address via a Neutron router, or via a suitable static route to the 169.254.169.254 address.
2. The metadata server knows about the virtual machine instance associated with the MAC and IP addresses of the virtual Ethernet interface issuing the metadata service request. This is necessary for the metadata server to be able to validate the request, and to identify the virtual machine's specific data to be returned.

On system managed subnets, the Neutron service has all address information associated with the virtual machines in its database.

On unmanaged subnets, you must tell the Neutron service the IP address of the network interface issuing the metadata service requests. This can only be done using the command line or API interfaces when instantiating the virtual machine, as illustrated below:

```
$ nova boot \
--nic net-id=net-uuid,vif-model=avp,v4-fixed-ip=172.18.0.1 \
my-new-vm
```

In this simplified example, the option v4-fixed-ip tells Neutron what the IP address for this interface should be. The interface's MAC address is automatically generated by Nova.

Guest VLANs

Use guest VLANs to segregate IP traffic from a single virtual Ethernet interface on a virtual machine into dedicated VLANs. Together with the capability to define multiple IP subnets on a single tenant network, guest VLANs facilitate the transitioning of existing guest applications to run on the HP Helion OpenStack Carrier Grade virtualized network environment.

Guest VLANs are useful when guest applications rely on the capability to configure a single virtual Ethernet interface with multiple, probably overlapping, IP addresses. From the point of view of the guest, this is done by defining VLAN Ethernet interfaces, and associating one or more IP addresses to them. If implementing overlapping IP addresses, typically in support of VPN applications, the guest must use different VLAN IDs to separate traffic from different VPNs.

For example. on a Linux guest, the virtual interfaces eth0.10:1, eth0.10:2, and eth0.20 refer to the same eth0 physical interface with two virtual interfaces on VLAN ID 10, and a single virtual interface on VLAN ID 20. A common scenario in a VLAN application is to allocate distinct IP addresses from the same IP subnet to virtual interfaces on the same VLAN. In this example, eth0.10:1 and eth0.10:2 could be assigned distinct IP addresses from the subnet 192.168.1.0/24, and eth0.20 an address from the subnet 192.168.2.0/24. In the case of a VPN application, overlapping IP addresses are allowed to exist on eth0.20 and either eth0.10:1 or eth0.10:2.

HP Helion OpenStack Carrier Grade supports these deployment scenarios with the help of guest VLANs which enable the transport of IP subnets traffic over VLAN-tagged Ethernet frames. To support the example above, a tenant user would define the following two IP subnets, both on the same tenant network, using guest VLAN IDs as follows:

- Subnet 192.168.1.0/24 with guest VLAN ID set to 10
- Subnet 192.168.2.0/24 with guest VLAN ID set to 20

The subnet-to-VLAN_ID mapping can be one-to-one, as in this example, or many-to-one. This means that tenant users can use a single VLAN ID of their choice to encapsulate traffic from one or more IP subnets.

Alternately, tenant networks can be configured to use VLAN Transparent mode, in which VLAN tagged guest packets are encapsulated within a provider network segment without removing or modifying the guest VLAN tag. For more information, see *VLAN Transparent* on page 44.

Creating Guest VLANs

Tenant users can define a guest VLAN when they add a new IP subnet to a tenant network they are creating, or to an existing one. See the *HP Helion OpenStack Carrier Grade Reference Deployment Scenarios* for examples of how to create tenant networks.

From the web management interface you add the VLAN ID of the guest VLAN to the **Subnet Detail** tab of the **Create Subnet** window. In the following example a guest VLAN with ID of 10 will be created when the subnet is added.

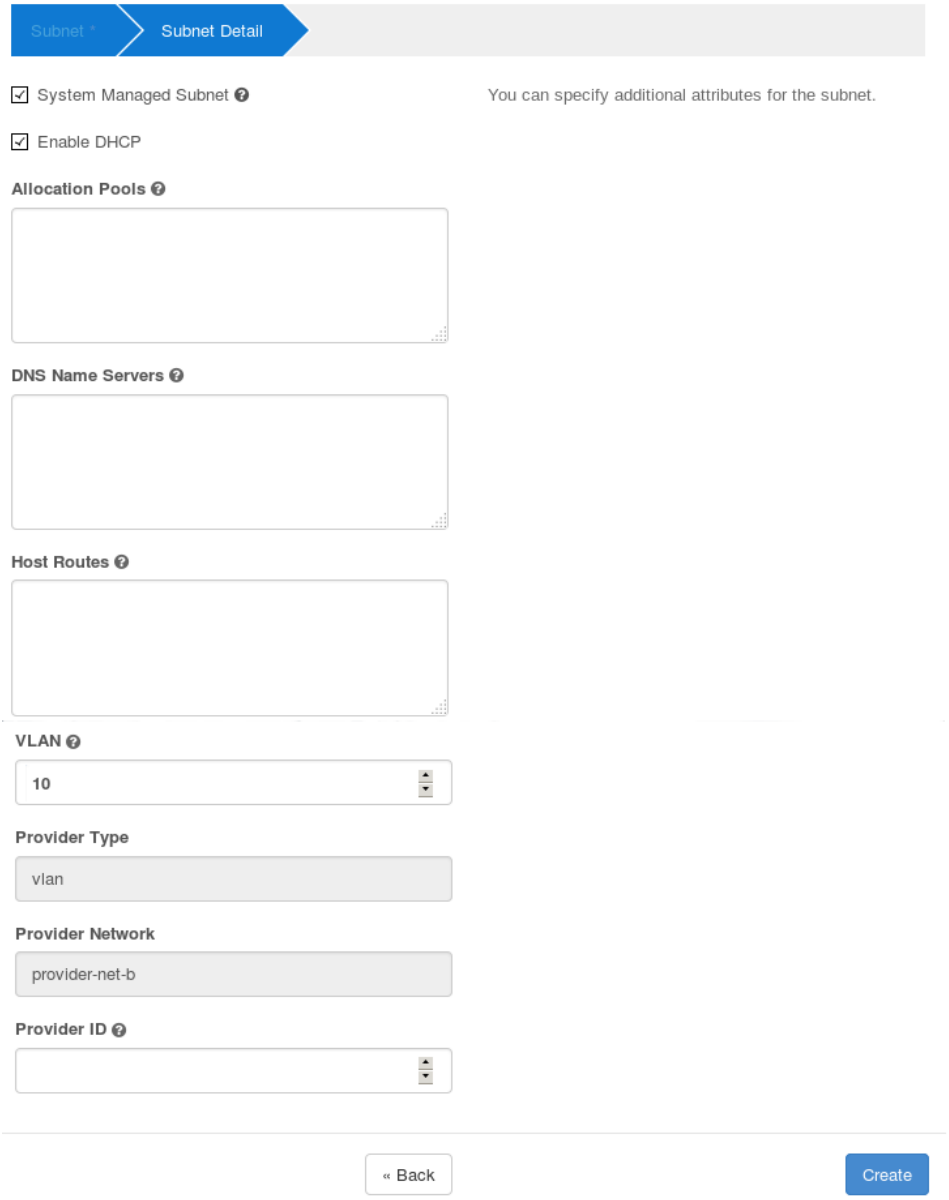


Figure 2: Creating a guest VLAN

From the command line interface, a guest VLAN can be defined using the --vlan-id option when creating a new subnet, as illustrated below:

```
~(keystone_admin)$ neutron subnet-create --tenant-id ${tenant_UUID} \
--name my-new-subnet --vlan-id 10
```

Guest VLAN Implementation

Guest VLANs are implemented using available segmentation ranges from suitable provider networks, just as it is done when new tenant networks are created. Therefore all network design considerations regarding the configuration of L2 switches and the Neutron allocation strategy, described in *Tenant Networks* on page 43, must be taken into consideration.

Additionally, note that the AVS will silently discard incoming VLAN-tagged VM traffic with unknown VLAN IDs, that is, with VLAN IDs not defined as guest VLANs on the particular tenant network.

L2 Access Switches

L2 access switches connect the HP Helion OpenStack Carrier Grade hosts to the different networks. Proper configuration of the access ports is necessary to ensure proper traffic flow.

One or more L2 switches can be used to connect the HP Helion OpenStack Carrier Grade hosts to the different networks. When sharing a single L2 switch you must ensure proper isolation of the network traffic. Here is an example of how to configure a shared L2 switch:

- one port- or MAC-based VLAN for the internal management network
- one port- or MAC-based VLAN for the OAM network
- one port-based VLAN for an optional board management network, if the network is configured for internal access
- one or more sets of VLANs for provider networks. For example:
  - one set of VLANs with good QoS for bronze tenants
  - one set of VLANs with better QoS for silver tenants
  - one set of VLANs with the best QoS for gold tenants

When using multiple L2 switches, there are several deployment possibilities. Here are some examples:

- A single L2 switch for the internal management and OAM networks. Port- or MAC-based network isolation is mandatory.
- One or more L2 switches, not necessarily inter-connected, with one L2 switch per provider network.
- Redundant L2 switches to support link aggregation, using either a failover model, or Virtual Port Channel (VPC) for more robust redundancy. For more information, see *Deploying Redundant Top-of-Rack Switches* on page 59.

Switch ports that send tagged traffic are referred to as *trunk* ports. They usually participate in the Spanning Tree Protocol (STP) from the moment the link goes up, which usually translates into several seconds of delay before the trunk port moves to the forwarding state. This delay is likely to impact services such as DHCP and PXE which are used during regular operations of HP Helion OpenStack Carrier Grade.

Therefore, you must consider configuring the switch ports to which the management interfaces are attached to transition to the forwarding state immediately after the link goes up. This option is usually referred to as a *PortFast*.

You should also consider configuring these ports to prevent them from participating on any STP exchanges. This is usually done by configuring them to avoid processing inbound and outbound BDPU STP packets completely. Consult your switch's manual for details.

Deploying Redundant Top-of-Rack Switches

For a system that uses link aggregation on some or all networks, you can configure redundant top-of-rack switches for additional reliability.

In a redundant ToR switch configuration, each link in a link aggregate is connected to a different switch, as shown in the accompanying figure. If one switch fails, another is available to service the link aggregate.

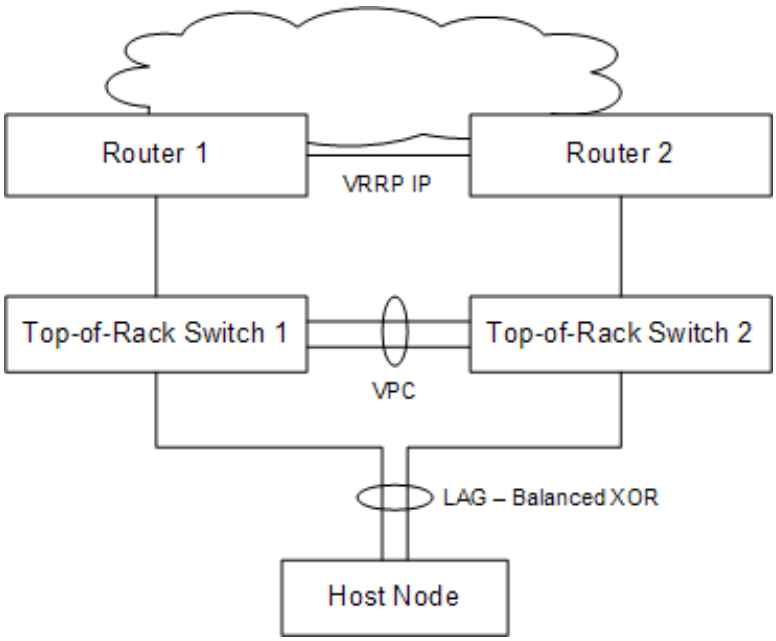


Figure 3: Redundant Top-of-Rack Switches

Switches that support Virtual Port Channel (VPC) are recommended. When VPC is used, the aggregated links on the switches act as a single LAG interface. Both switches are normally active, providing full bandwidth to the LAG. If there are multiple failed links on both switches, at least one connection in each aggregate pair is still functional. If one switch fails, the other continues to provide connections for all LAG links that are operational on that switch. For more about configuring VPC, refer to your switch documentation.

You can also use an active/standby failover model for the switches, but at some cost to overall reliability. If there are multiple failed links on both switches, then optimally the switch with the greatest number of functioning links is activated, but some links on that switch could be in a failed state. In addition, when only one link in an aggregate is connected to an active switch, the LAG bandwidth is limited to the single link.

Note:

You can further enhance system reliability by using redundant routers. For more information, refer to your router documentation.

DiffServ-Based Priority Queuing and Scheduling

Differentiated Services, or DiffServ, is a packet-based traffic management mechanism that allows end devices to specify an expected per-hop behavior (PHB) from upstream switches and routers when handling their traffic during overload conditions.

DiffServ marking and architecture are IEEE specifications, IEEE RFC 2474 and RFC 2475.

Support for DiffServ is implemented on the AVS for all input and output ports, on all attached tenant networks. On each port, it uses eight queues associated with the CS0 to CS7 DiffServ class selectors, which are processed by a round-robin scheduler with weights of 1, 1, 2, 2, 4, 8, 16, and 32. Overflow traffic is tail-drop discarded on each queue. Guest applications in the HP Helion OpenStack Carrier Grade cluster can set a Differentiated Services Code Point (DSCP) value in the Differentiated Service (DS) field of the IP header to mark the desired upstream PHB.

On ingress, a packet being processed to be sent to the VM is directed to the appropriate DiffServ output queue. On overflow, that is, if the virtual machine cannot keep up with the incoming traffic rate, lower priority packets are discarded first.

On egress, a packet sent from the VM to the AVS is first enqueued into the appropriate DiffServ input queue according to the specified DSCP value, the CS0 queue being the default. On overload, that is, if the AVS cannot keep up with the incoming traffic, lower priority packets are discarded first; in this case, you should consider re-engineering the AVS, possibly assigning it more processing cores. Once serviced by the DiffServ class scheduler, the

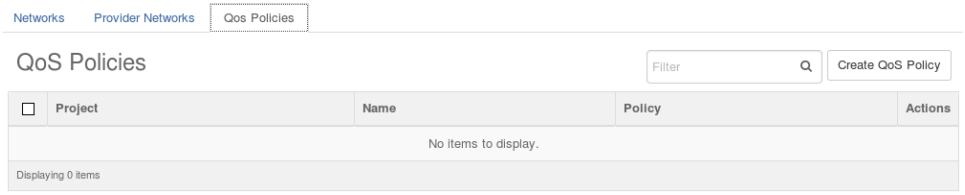
packet is processed according to the QoS policy in place for the tenant network, as described in *Quality of Service Policies* on page 61.

Quality of Service Policies

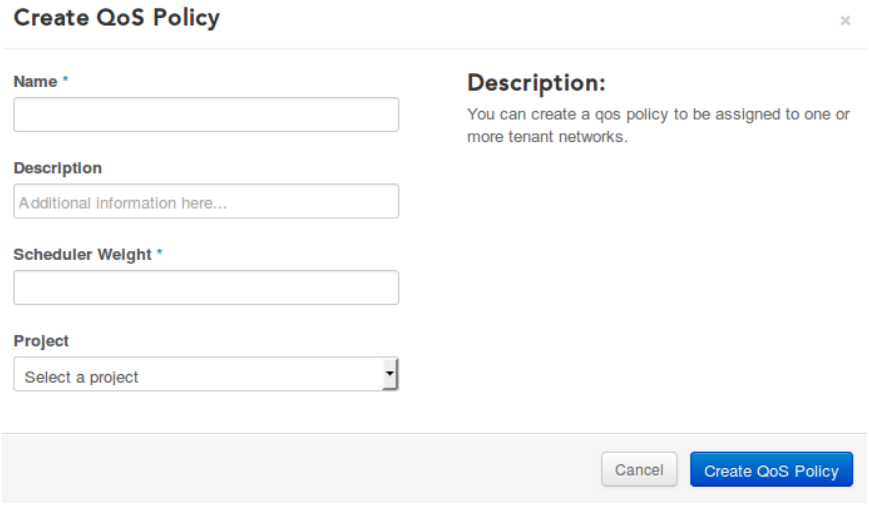
Quality of Service (QoS) policies specify relative packet processing priorities applied by the AVS switch on each compute node to incoming tenant network's traffic during overload conditions.

The QoS polices play no role under normal traffic loads, when no input traffic queues in the AVS are close to their overflow limits.

QoS policies are created by the cluster administrator, and selected by the tenant users to apply on a per-tenant network basis. To create a new QoS policy, log in as administrator, select the option Networks from the **Admin** menu, and visit the **QoS Policies** tab on the **Networks** page, as illustrated below.



Click the button **Create QoS Policy** to display the **Create QoS Policy** window.



The following fields are available:

Name

The name of the new QoS policy. This is the name that the tenant user sees as available for use.

This field is mandatory.

Description

A free form text for your reference.

Scheduler Weight

The relative weight the AVS traffic scheduler uses on overload conditions, as compared with the scheduler weight of all other QoS policies.

The scheduler weight is a positive integer number associated with each QoS policy. On overload, the AVS schedules traffic processing for each tenant network according to its assigned QoS policy. By default, with no specific QoS policies defined, traffic from all tenant networks is processed in round-robin mode, one tenant network after another. Effectively, the default behavior is similar to assigning a scheduler weight of 1 to all tenant networks.

When a specific QoS policy with a scheduler weight greater than 1 is applied to a tenant network, its traffic is scheduled with a relative higher frequency. For example, if the scheduler weight for tenant network A is 10, and the one for tenant network B is 100, then on overload, tenant network B will see its queued traffic processed 10 times as often as tenant network A.

The handling of the scheduler weight is implemented as a per-packet token bucket for each tenant network. This means that each unit in the scheduler weight counts as one packet to be processed in sequence. In the previous example, the AVS processes 10 consecutive packets from tenant network A, followed by 100 packets from tenant network B. This implementation ensures a fair-share behavior that prevents any tenant network from running into total bandwidth starvation, even if its scheduler weight is relatively low.

The range of values for the scheduler weight is arbitrary. You must however ensure that the values assigned to the different policies make sense for the intended applications.

This field is mandatory.

Project

A drop-down menu displaying the currently defined tenants (projects). The new QoS policy is available to only the selected tenant. If no tenant is selected, it is available to all tenants in the cluster.

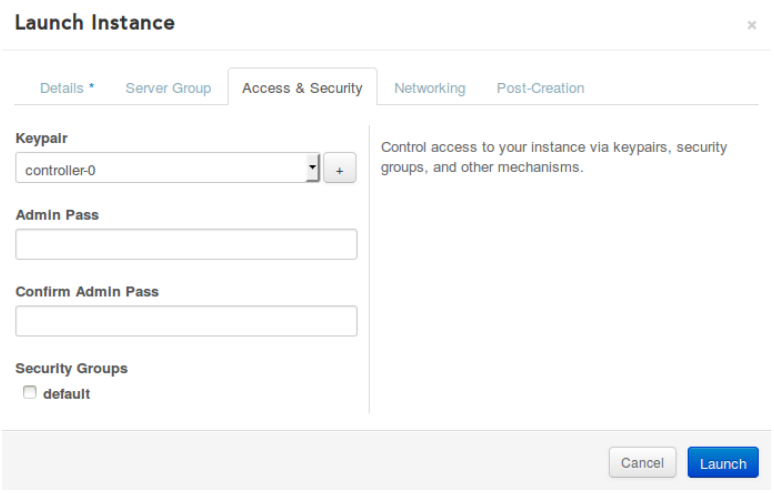
Tenant users can select available QoS policies when the tenant networks are created from the **Create Network** window. They can also apply a QoS policy to an already existing tenant network from the **Edit Network** window.

Security Groups

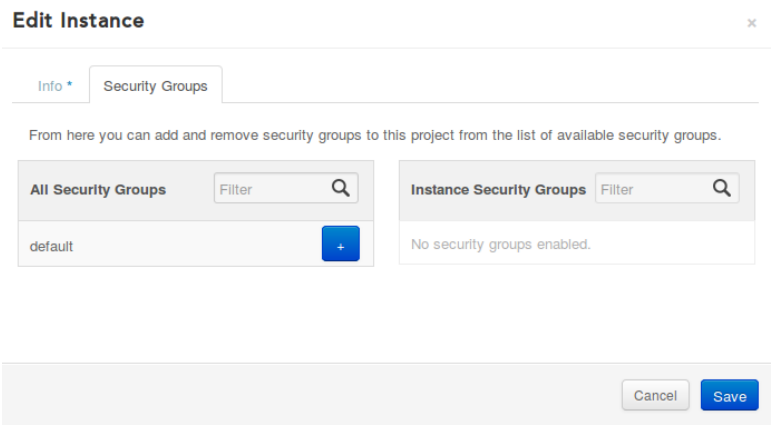
Security groups are tenant-specific sets of IP filter rules that are applied to the networking stack of a virtual machine.

HP Helion OpenStack Carrier Grade does not enforce a default security group when launching a new virtual machine. Instead, the security groups are optional, and only enforced when explicitly assigned to a virtual machine, either at launch time, or while it is executing.

At launch time, select the desired security groups from the tab **Access & Security** on the **Launch Instance** window, as illustrated below.



To assign security groups to a running instance, display the **Instances** page, click the **More** button associated with the instance, and select the option Edit Security Groups. The **Edit Instance** window displays with the tab **Security Groups** showing the available security instance groups, as illustrated below.



You can then drag and drop security groups from the section **All Security Groups** to the section **Instance Security Groups**. The selected security groups are applied immediately once the configuration is saved.

In the HP Helion OpenStack Carrier Grade implementation, the following limitations apply:

- the maximum number of security groups that can be applied to a single tenant network is 10
- the maximum number of rules that can be configured on a single tenant network is 100

These two limitations work always together, whichever is first reached.

REST API Secure Access Planning

For secure REST API access, you must obtain and install a signed digital certificate before software installation.

HP Helion OpenStack Carrier Grade supports programming access using REST APIs. By default, access using the HTTP protocol is supported. For increased security, you can enable HTTPS access during system installation.

To enable HTTPS access, you must obtain a digital certificate and copy it to the first controller before starting the controller configuration script. For more information, see the *HP Helion OpenStack Carrier Grade Software Installation Guide*.

Storage Planning

Storage resources on the controller nodes are used to maintain internal databases, and to provide storage for virtual machines. For VMs, it is highly recommended to use storage provided through volumes (Cinder service backed by 3Par). There might not be enough storage on compute nodes (especially blades).

During HP Helion OpenStack Carrier Grade software installation, the controller configuration script gives you the option to use the controller node or dedicated storage nodes to provide storage. The storage is used for the internal database, and for general storage volumes (images and disk space for the virtual machines).

If the controller node is used, the storage is LVM-based. If storage nodes are used, the storage is 3PAR-based.

The size of the database grows with the number of system resources created by the system administrator and the tenants. This includes objects of all kinds such as compute nodes, provider networks, images, flavors, tenant networks, subnets, virtual machine instances and NICs. As a reference point, consider the following deployment scenario:

- two controllers
- four compute nodes with dual Intel(R) Xeon(R) CPU E5-2670 v2 @ 2.50GHz each.
- 40 virtual machine instances
- 120 tenant networks
- steady collection of power management statistics

The size of the database in this case is approximately 9 GB. With a suggested default of 20 GB, there is still plenty of room to grow. However, you should periodically monitor the size of the database to ensure that it does not become a bottleneck when delivering new services.

For general storage, the default settings suggested during the controller configuration script are recommended to utilize the maximum available space on the storage media.

Storage for Virtual Machines

For VMs, you can use remote Cinder storage or local storage.

Persistent block storage for virtual machines is allocated by the Cinder service using 3PAR.

LVM/iSCSI

This backend provides block storage managed by the Linux Logical Volume Manager (LVM) exposed as iSCSI targets. In the HP Helion OpenStack Carrier Grade implementation:

- Storage space is allocated on the active controller and automatically mounted over iSCSI by the virtual machines running on the compute nodes. No storage space is allocated on the compute nodes.
- The two controllers always maintain their storage partitions in sync for carrier-grade reliability.
- This backend does not support storage nodes.

The LVM/iSCSCI and 3PAR-based storage backend options are exclusive. You select one or another when configuring **controller-0** by selecting the appropriate storage option.

As an alternative to persistent storage provided by the Cinder service, you can implement ephemeral local storage on the compute nodes where the VMs are instantiated. This is useful for VMs requiring local disk access for performance optimization. You can use a pre-allocated partition on the root disk, as well as additional disks optionally installed in the compute nodes. For more information, see *Configuring a Compute Host to Provide Local Storage* on page 64.

Caution:

Local storage is ephemeral.

- Unlike Cinder-based storage, local storage does not persist if the instance is terminated or the compute node fails.
- Live migration is not currently supported for an instance using local storage. Locking the host initiates a cold migration. The local storage for the instance is rebuilt.
- Resizing of local storage is partly supported. If the reconfigured instance is instantiated on the same host, then any root or ephemeral disks that use local storage are resized with their data preserved. Swap disks that use local storage are rebuilt. If the instance is migrated to another host, only cold migration is supported. All local storage for the instance is rebuilt.

To instantiate VMs on compute nodes configured for local storage, see *Specifying Local Storage for VM Ephemeral Resources* on page 153.

Configuring a Compute Host to Provide Local Storage

You can configure a compute host to provide local storage for use by VMs.

Local storage can be used to provide improved disk access performance for VMs instantiated on the host.

Caution:

Local storage is ephemeral.

- Unlike Cinder-based storage, local storage does not persist if the instance is terminated or the compute node fails.
- Live migration is not currently supported for an instance using local storage. Locking the host initiates a cold migration. The local storage for the instance is rebuilt.
- Resizing of local storage is partly supported. If the reconfigured instance is instantiated on the same host, then any root or ephemeral disks that use local storage are resized with their data preserved. Swap

disks that use local storage are rebuilt. If the instance is migrated to another host, only cold migration is supported. All local storage for the instance is rebuilt.

To provide local storage on a compute host, add a local volume group called **nova-local** to the host. This automatically includes the host in the **local_storage_hosts** host aggregate. The Nova scheduler draws from this host aggregate to instantiate VMs that require local storage.

To populate the **nova-local** group with storage resources, you must also add physical volumes created on local disks.

You must manually allocate space on the local volume group for an instances logical volume. For more information, see *Instances Logical Volume Considerations* on page 69.

Caution:

If less than the minimum required space is available for the instances logical volume, the compute host cannot be unlocked.

As an alternative to the web administration interface, you can use the CLI to configure the host for local storage. For more information, see *Managing Local Volume Groups* on page 128 and *Managing Physical Volumes* on page 129.

1. Lock the host to make changes.
   a) On the **Admin** menu of the web administration interface, open the **System Panel** section, and then select **Inventory**.
   b) Select the **Hosts** tab.
   c) Open the **More** drop-down list for the host, and then select **Lock Host**.
   d) Wait for the host to be reported as **Locked**.
2. Open the **Inventory Detail** page for the locked host.

   In the **Host Name** column, click the name of the host.
3. Select the **Storage** tab.

   For a compute node, the storage tab displays three lists:

   Disks

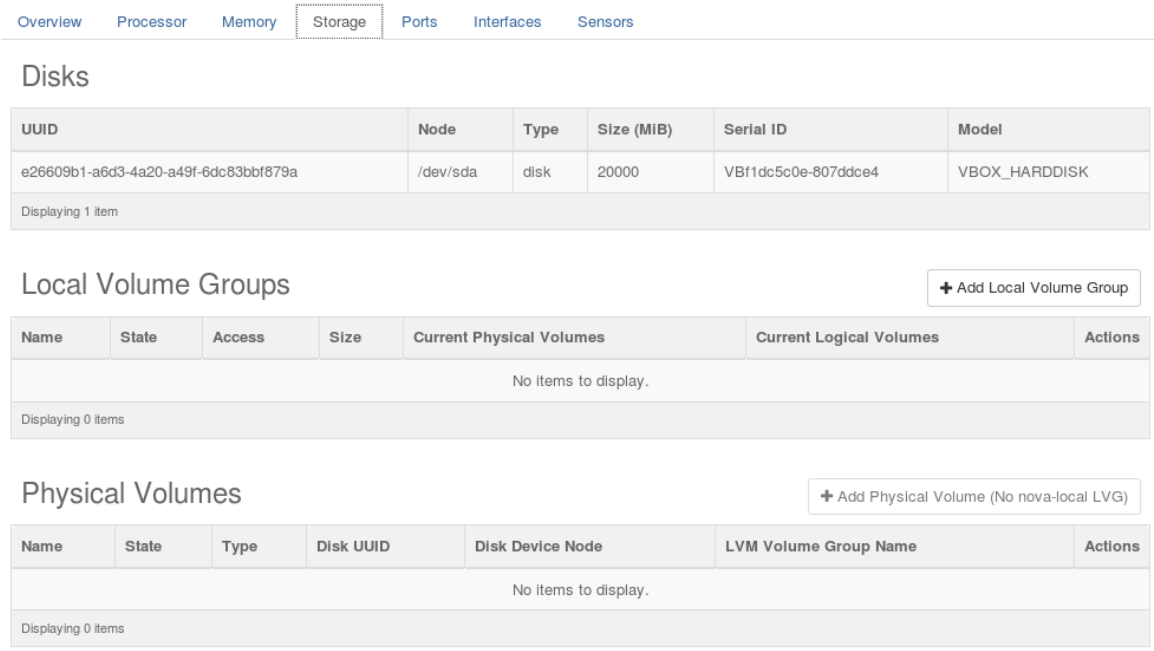
   Physical disks installed in the compute node.

   Local Volume Groups

   Groups of physical volumes available for VM storage.

   Physical Volumes

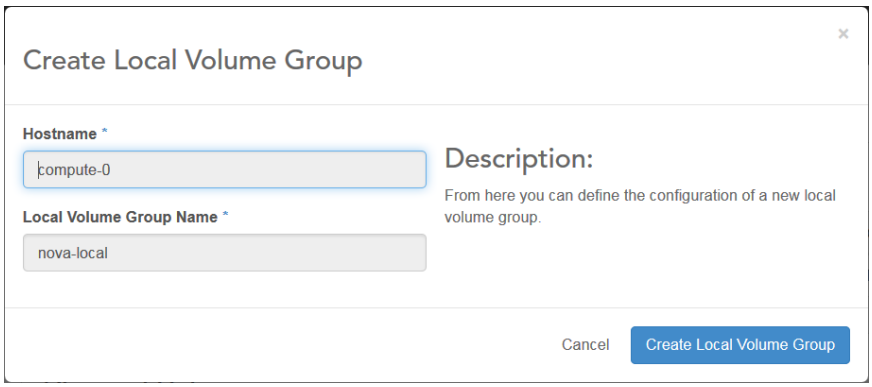
   Partitions that have been initialized for logical volume management (LVM) and assigned to a local volume group.



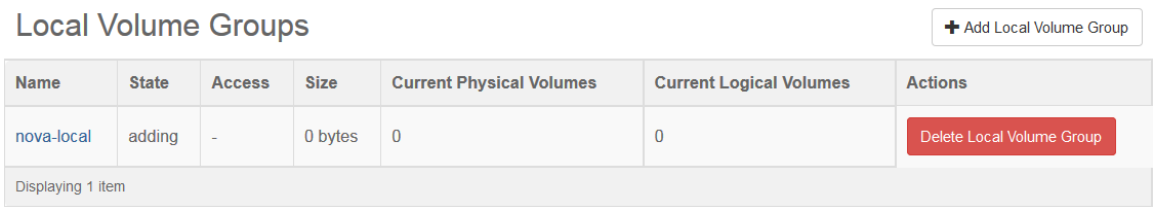
4. If required, add the **nova-local** group to the **Local Volume Groups** list.

   The **Local Volume Groups** list displays groups of physical volumes designated for use as local storage.

   The default group for use by VMs is called **nova-local**. If this group is not listed, you can create it by clicking **Add Local Volume Group**, and then accepting the defaults in the dialog box that appears.

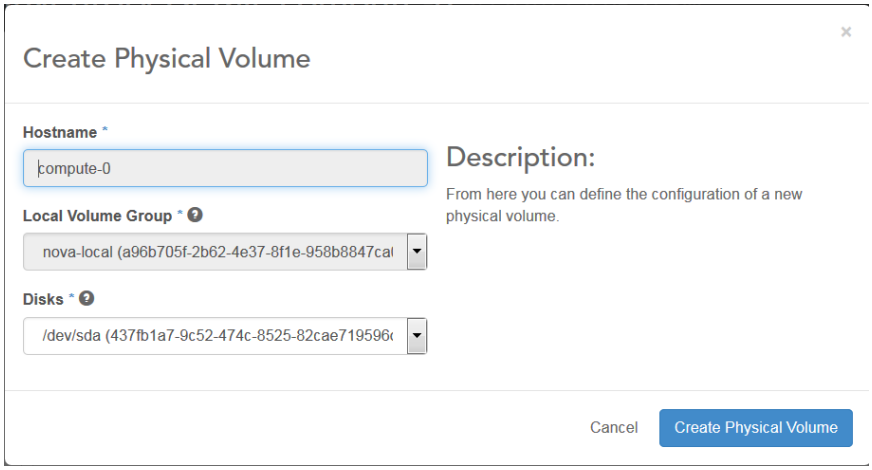


   The **nova-local** group is shown in the list.



5. Add a physical volume to the **nova-local** group.

   The **Physical Volumes** list displays LVM volumes that have been created and added to a Local Volume Group. To add a new one, click **Add Physical Volume** and complete the dialog box that appears.



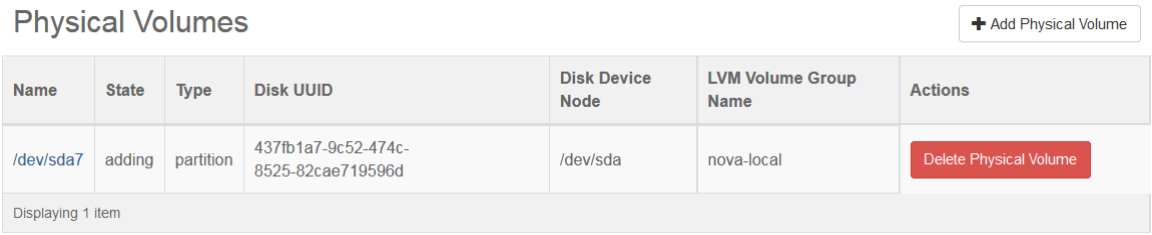
Note:

The **Local Volume Group** drop-down selection is automatically set to **nova-local**, and cannot be changed.

Use the **Disks** drop-down menu to select a disk to provide storage. For the root disk, a designated partition (**local_pv**) is used.

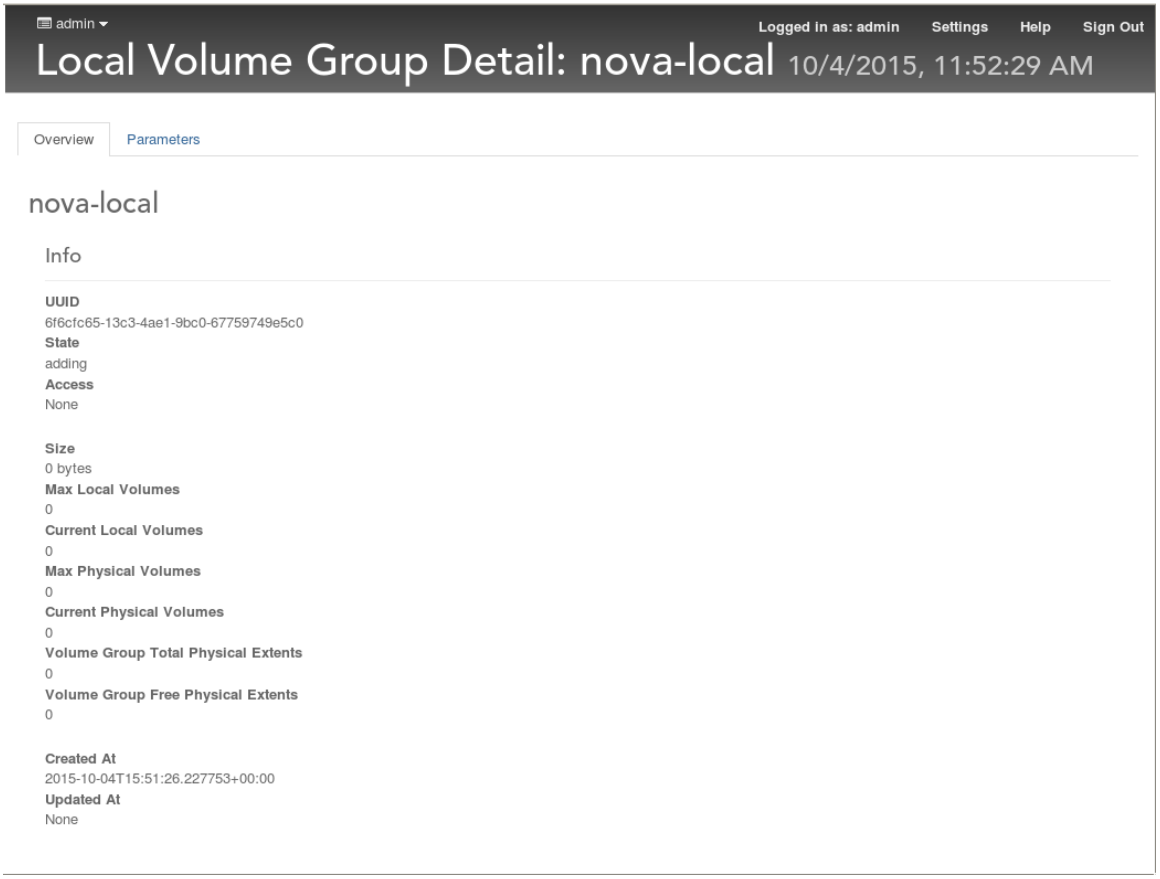
You can add any number of physical volumes to the **nova-local** group, limited only by the number of available disks.

A physical volume is created and added to the **nova-local** group.

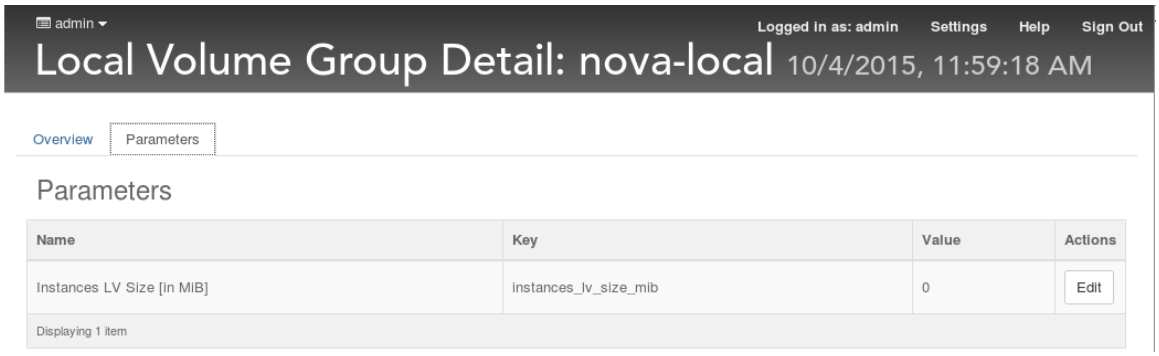


6. Allocate space for the instances logical volume.

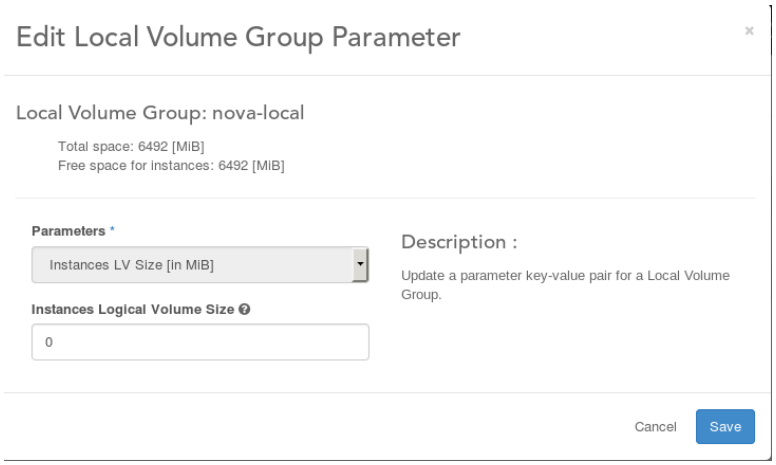
   In the **Local Volume Groups** list, click **nova-local** to open the Local Volume Group Detail page.



Select the **Parameters** tab.



Click **Edit**, and then set the **Instances Logical Volume Size**.



If the total space in the volume group is less than 80GB (81920 MiB), then you must allocate at least 2048 MiB. If the volume group is greater than 80GB (81920 MiB), then you must allocate at least 5120 MiB. The actual size is limited to ensure at least 50% free space in the volume group for creating disks for launched instances. For more information, see *Instances Logical Volume Considerations* on page 69

7. Unlock the host to make the changes take effect.

The compute host is ready to provide local storage for use by VMs. Because a **nova-local** local volume group has been added, the host is automatically included in the **local_storage_hosts** host aggregate.

To return a host to the **remote_storage_hosts** host aggregate, so that it is used to instantiate VMs configured for remote Cinder-based storage, delete the **nova-local** local volume group.

Caution:

VMs instantiated on the compute node must be migrated or terminated before the host is reconfigured for remote storage.

To configure a VM to use local storage or remote Cinder-based storage, or to be instantiated on any compute host regardless of storage type, see *Specifying Local Storage for VM Ephemeral Resources* on page 153.

To add disks to a compute node for extra local storage, see *Adjusting Resources on a Compute Node* on page 115

Instances Logical Volume Considerations

On compute nodes that use local storage, dedicated space is required by the **nova** service.

The space is needed to store files that track each launched instance, and to cache boot images for boot-by-image VMs. Boot images are always cached in RAW format; Glance images formatted as QCOW2 are expanded to RAW format. The use of a cache speeds up instance launch, because the image does not need to be fetched from the Glance image store.

In HP Helion OpenStack Carrier Grade, the space is called the *instances logical volume* or **Instances LV** space. To provide for scalability, it is located on the **nova-local** local volume group instead of the root partition.

By default, no size is specified. The minimum required space is 2 GB for a volume group with a total size less that 80 GB, and 5 GB for a volume group larger than 80 GB; you must specify at least this amount. You can allocate more to support the anticipated number of boot-from-image VMs, up to 50% of the maximum available storage for the local volume group. At least 50% free space in the volume group is required to provide space for allocating logical volume disks for launched instances. The value provided for the **Instance LV Size** is limited by this maximum.

Instructions for allocating the **Instances LV Size** using the web administration interface are included in *Configuring a Compute Host to Provide Local Storage* on page 64. To allocate space using the CLI, use the `system host-lvg-modify` command.

Caution:

If less than the minimum required space is available, the compute host cannot be unlocked.

Using VXLANs

You can use Virtual eXtensible Local Area Networks (VXLANs) to connect VM instances across non-contiguous Layer 2 segments (that is, Layer 2 segments connected by one or more Layer 3 routers).

A VXLAN is a Layer 2 overlay network scheme on a Layer 3 network infrastructure. Packets originating from VMs and destined for other VMs are encapsulated with IP, UDP, and VXLAN headers and sent as Layer 3 packets. The IP addresses of the source and destination compute nodes are included in the headers.

You can configure VXLANs on HP Helion OpenStack Carrier Grade using the following workflow.

1. Set up a provider network of the VXLAN type.

   For details, see *Setting Up a VXLAN Provider Network* on page 70.
2. Configure the endpoint IP addresses of the compute nodes.

For details, see *Configuring Endpoint IP Addresses Using the CLI* on page 77.

3. Establish routes between the hosts.

   For details, see *Adding and Maintaining Routes for a VXLAN Network* on page 78

You must also ensure that the networking environment meets certain minimum requirements. For more information, see *Tenant Networks* on page 43.

Setting Up a VXLAN Provider Network

You can use the CLI or the web administration interface to set up a VXLAN provider network and add segmentation ranges.

VXLAN provider networks are an alternative to VLAN provider networks when VM L2 connectivity is required across separate Layer 2 network segments separated by one or more Layer 3 routers.

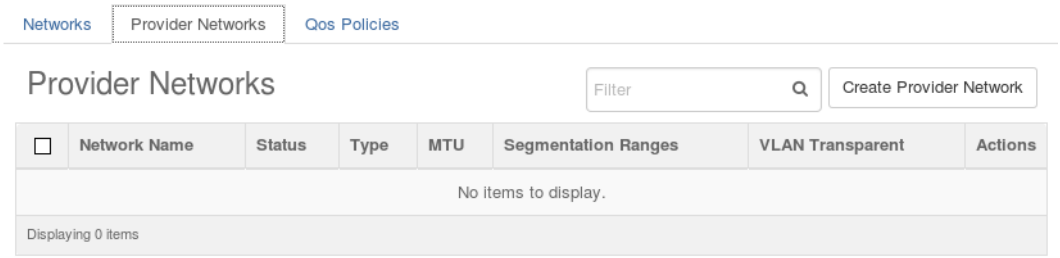
The steps in this section describe how to set up a VXLAN provider network and add segmentation ranges using the web administration interface. For information about using CLI commands, see *Setting Up a VXLAN Provider Network Using the CLI* on page 72.

1. Open the HP Helion OpenStack Carrier Grade web administration interface.

   Using a browser, navigate to the OAM floating IP address, and log in as **admin**.

2. In the left-hand pane, select **Admin** > **System** > **Networks**, and then select the **Provider Networks** tab.

   The Provider Networks list is displayed.



3. Create a provider network.

   Click **Create Provider Network**.

   In the **Create Provider Network** window, complete the fields as required.

   Name

   The name of the provider network.

   Description

   A free-text field for reference.

   Type

   The type of provider network to be created.

   flat

   mapped directly to the physical network

   vlan

   supports multiple tenant networks using VLAN IDs.

   vxlan

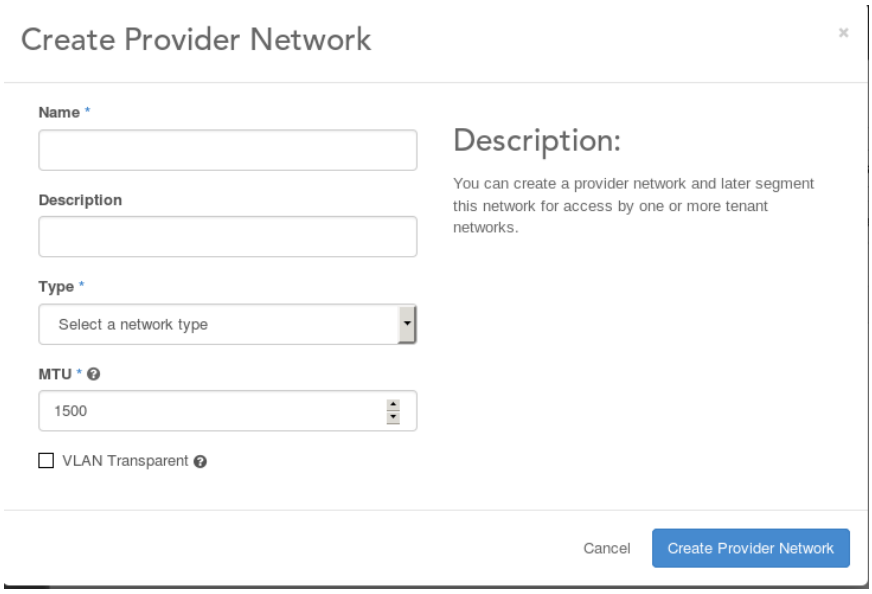
   supports multiple tenant networks using VXLAN VNIs.

MTU

The maximum transmission unit for the Ethernet segment used to access the network.

VLAN Transparent

Allow VLAN tagged packets to be encapsulated within a VXLAN segment without removing or modifying the guest VLAN tag.



For the **Type**, select **VXLAN**.

For the MTU, set a maximum transmission unit that allows for the overhead required by VXLAN encapsulation. For more information, see *The Ethernet MTU* on page 26.

4. Commit the changes.

   Click **Create Provider Network**.

   The new provider network is added to the **Provider Networks** list.

5. Add one or more segmentation ranges.

   Segmentation ranges are sets of contiguous identifiers defined on a provider network. Each ID is used to implement a tenant network.

   On the **Provider Network Overview** page, click **Create Range** to open the **Create Segmentation Range** page. Complete the form as required.

   Name

   The name of the segmentation range.

   Description

   A free-text field for reference.

   Shared

   If selected, shares the range for use by all tenants.

   Project

   The tenant associated with the segmentation range.

   Minimum

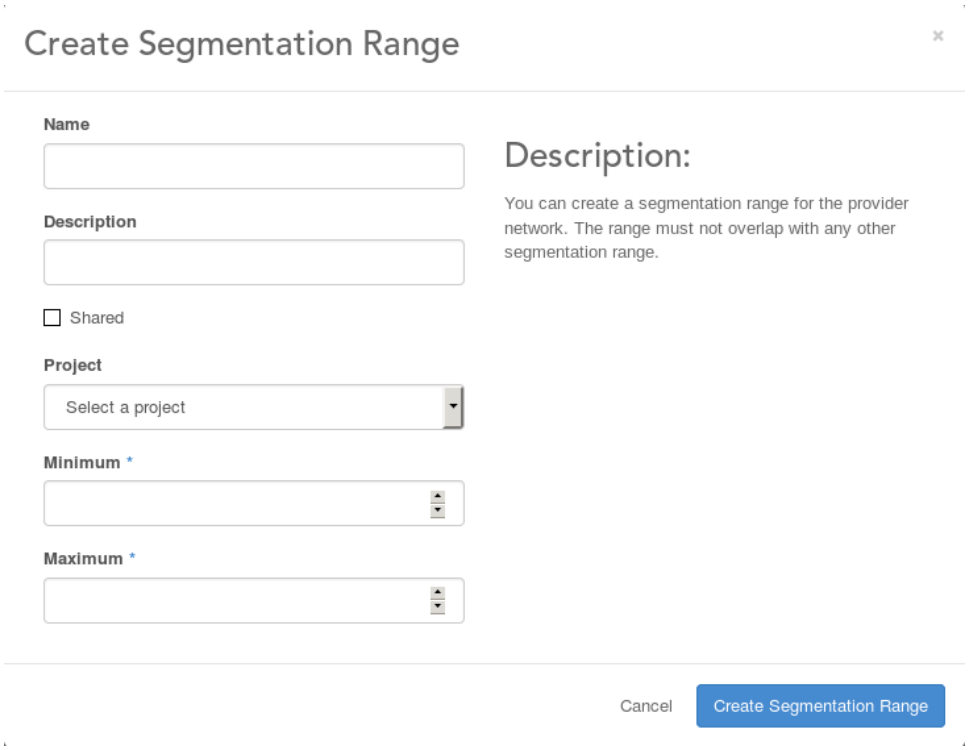
   The lowest value of a range of IDs.

Maximum

The highest value of a range of IDs.

Caution:

The range must not overlap other segmentation ranges on the same provider network.



6. Click **Create Segmentation Range** to commit the changes.

Setting Up a VXLAN Provider Network Using the CLI

You can use the command line interface to set up a VXLAN provider network and add segmentation ranges.

VXLAN provider networks are an alternative to VLAN provider networks when VM L2 connectivity is required across separate Layer 2 network segments separated by one or more Layer 3 routers.

The steps in this section describe how to set up a VXLAN provider network and add segmentation ranges using the command line interface. For information about using the web administration interface, see *Setting Up a VXLAN Provider Network* on page 70.

To create a provider network using the CLI, use the following command:

```
~(keystone_admin)$ neutron providernet-create name \
--type=type --description=description mtu mtu_size \
--vlan-transparent={True,False}
```

where

name

is a name for the provider network

type

is the type of provider network (**flat**, **vlan**, or **vxlan**)

description

is a brief description for reference purposes

mtu_size

is the maximum transmission unit size

To add a segmentation range using the CLI, use the following command:

```
~(keystone_admin)$ neutron providernet-range-create provider_network \
--name=range_name --tenant_id=tenant \
--description=description --range min-max \
--group multicast_address --port=udp_port \
--ttl=time_to_live
```

where

provider_network

is the name of the associated provider network

name

is a name for the segmentation range

tenant

is the name or UUID of the tenant associated with the range

description

is a brief description for reference purposes

min

is the lowest value in the range

max

is the highest value in the range

vlan-transparent

Allow VLAN tagged packets to be encapsulated within a VXLAN segment without removing or modifying the guest VLAN tag.

The following additional values are used for segmentation ranges on VxLAN provider networks:

multicast_address

The IPv4 or IPv6 address for participation in a multicast group used to discover MAC addresses for destination VMs. You can use a different multicast group for each segmentation range to help organize and partition network traffic.

udp_port

The destination UDP port for packets sent on the VXLAN. You can select either the IANA standard 4789 to use this range with the OpenStack Neutron service, or the legacy standard 8472 for use with some commercial switch equipment.

time_to_live

The time-to-live, measured in hops, for packets sent on the VXLAN. The value is decremented at each hop; when it reaches zero, the packet expires. You can use this to limit the scope of the VXLAN. For example, to limit the packet to no more than three router hops, use a time-to-live value of 4.

For more about these parameters, refer to the steps for using the web administration interface.

You can obtain information about provider networks and segmentation ranges using the following commands.

```
~(keystone_admin)$ neutron net-list-on-providernet providernet
```

```
~(keystone_admin)$ neutron providernet-range-show providernet-range
```

Adding an IP Address to a Data Interface

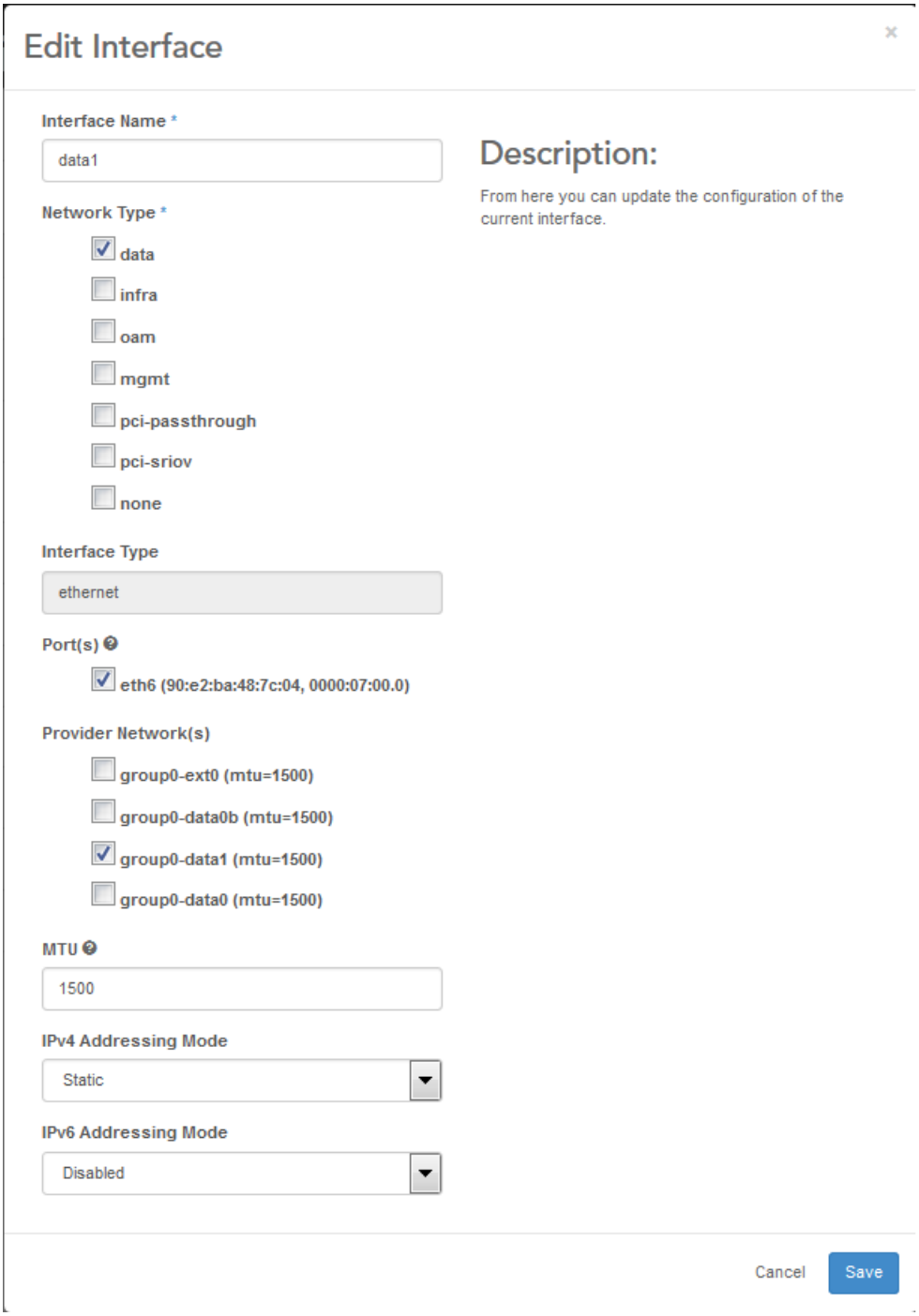
You can add IP addresses to a data interface using the web administration interface or the command line.

For VXLAN connectivity between VMs, you must add appropriate endpoint IP addresses to the compute node interfaces.

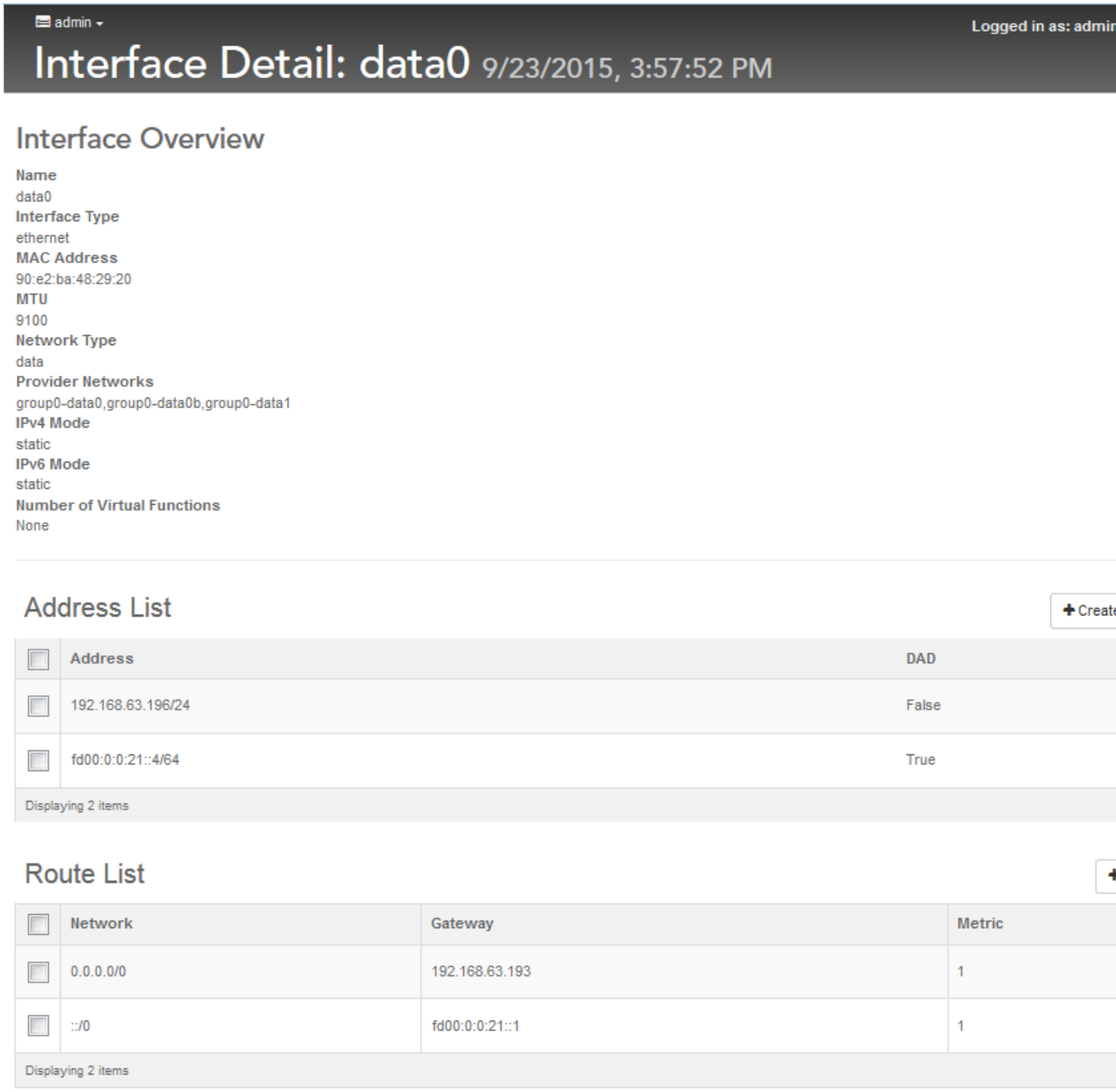
To add an IP address using the web administration interface, refer to the following steps. To use the command-line interface; see *Configuring Endpoint IP Addresses Using the CLI* on page 77

To make interface changes, you must lock the compute host first.

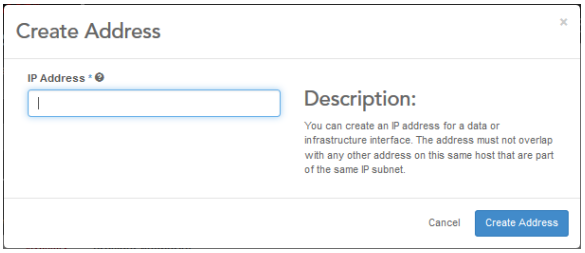
1. Lock the compute host.
2. Set the interface to support an IPv4 or IPv6 address, or both.
   a) On the **Admin** menu, click **Inventory** to open the **System Inventory** page.
   b) Select the **Host** tab, and then double-click the compute host to open the **Host Detail** page.
   c) Click **Edit Interface** for the data interface you want to edit.
   d) In the **Edit Interface** dialog box, set the **IPv4 Addressing Mode** or the **IPv6 Addressing Mode** to **Static**.



3. Add an IPv4 or IPv6 address to the interface.
   a) On the **Inventory Detail** page, click the **Name** of the interface to open the **Interface Detail** page.



b) Click **Create Address** to open the **Create Address** dialog box.



c) Enter the IPv4 or IPv6 address and netmask (for example, 192.168.1.3/24), and then click **Create Address**.

The new address is added to the **Address List**.

4. Unlock the compute node and wait for it to become available.

Configuring Endpoint IP Addresses Using the CLI

You can configure a data interface on a compute node with an endpoint address using CLI commands.

For VXLAN connectivity between VMs, the compute nodes that host the VMs must be configured with endpoint IP addresses.

To make interface changes, you must lock the compute node first.

1. Lock the compute node.
2. Set the interface to support an IPv4 or IPv6 address, or both.

```
~(keystone_admin)$ system host-if-modify node ifname --ipv4-mode=ipv4mode
 --ipv6-mode=ipv6mode
```

where

node

is the name or UUID of the compute node

ifname

is the name of the interface

ipv4mode

is either **disabled** or **static**

ipv6mode

is either **disabled** or **static**

3. Add an IPv4 or IPv6 address to the interface.

```
~(keystone_admin)$ system host-addr-add node ifname ip_address prefix
```

where

node

is the name or UUID of the compute node

ifname

is the name of the interface

ip_address

is an IPv4 or IPv6 address

prefix

is the netmask length for the address

To delete an address, use the following commands:

```
~(keystone_admin)$ system host-addr-list
```

This displays the UUIDs of existing addresses.

```
~(keystone_admin)$ system host-addr-delete uuid
```

where **uuid** is the UUID of the address.

4. Unlock the compute node and wait for it to become available.

Adding and Maintaining Routes for a VXLAN Network

You can add or delete routing table entries for hosts on a VXLAN network using the CLI.

The compute node must be locked.

To add routes, use the following command.

```
~(keystone_admin)$ system host-route-
add node ifname network prefix gateway metric
```

where

node

is the name or UUID of the compute node

ifname

is the name of the interface

network

is an IPv4 or IPv6 network address

prefix

is the netmask length for the network address

gateway

is the default gateway

metric

is the cost of the route (the number of hops)

To delete routes, use the following command.

```
~(keystone_admin)$ system host-route-
delete uuid ifname network prefix gateway metric
```

where **uuid** is the UUID of the route to be deleted.

To list existing routes, including their UUIDs, use the following command.

```
~(keystone_admin)$ system host-route-list compute-0
```

Chapter 3

System Configuration Management

Topics:

You can make changes to the initial HP Helion OpenStack Carrier Grade configuration at any time after installation.

The initial configuration is defined during software installation, using the **config_controller** script (see the *HP Helion OpenStack Carrier Grade Software Installation Guide: Configuring Controller-0).* After installation, you can make changes using the web administration interface or the CLI. For most types of changes, you can do this without interrupting system operations.

You can change the initial configuration as follows:

- Reconfigure aspects of the External OAM network, including the subnet, controller IP addresses, gateway IP address, and floating IP address.
- Change the DNS server IP addresses.
- Change the NTP server IP addresses.
- Change the disk allocations for database storage, image storage, backup storage, and Cinder volume storage.
- Add an infrastructure network if one is not already configured.

  Note:

  A service interruption is required to add an infrastructure network; all nodes except the active controller must be locked. In addition, you must use the CLI to make this change.

Immediately before making a change, ensure the following:

- All hosts are unlocked.
- No **Major** or **Critical** alarms are shown for the system on the Fault Management page.
- No **250.001 Configuration out-of-date** alarms are shown for the system on the Fault Management page.

During these changes, HP Helion OpenStack Carrier Grade can remain online and operational. For some types of changes, alarms are raised while the settings are in transition. To make the changes take effect and to clear the alarms, individual hosts must be locked and then unlocked one at a time. For more information, see *Host Status and Alarms During System Configuration Changes* on page 81.

Only hosts with **250.001 Configuration out-of-date** alarms need to be locked and then unlocked. When locking and unlocking the hosts to clear the alarms, use the following order:

1. controller nodes
2. compute nodes

Note:

To keep the system operational during an OAM network change, sufficient resources must be available to migrate the virtual machines on a compute node while it is locked for changes.

To make changes using the web administration interface, you must be logged in as the **admin** user. Use the **System Configuration** pane, is available from the **System Panel** section of the **Admin** menu.

To make changes using the CLI, you must be logged in as the **wrsroot** user on the active controller, and sourced as the Keystone **admin** user. For more information, see *System Configuration Management Using the CLI* on page 88.

Host Status and Alarms During System Configuration Changes

For all types of configuration changes, alarms and status messages appear while the system is in transition. You can use the information provided by these messages to help guide the transition successfully.

Configuration changes require multiple hosts to be reconfigured, during which the settings across the cluster are not consistent. This causes alarms to be raised for the system.

- Changes to the DNS server configuration cause transitory alarms. These alarms are cleared automatically when the configuration change is applied.
- Changes to the External OAM network IP addresses or NTP server addresses, and in particular to the controller storage allotments, cause persistent alarms. These alarms must be cleared manually, by locking and unlocking the affected hosts or performing other administrative actions.

Alarms appear on the **Fault Management** pane, and related status messages appear on the **Hosts** tab on the **Inventory** pane. A variety of alarms may be reported on the Fault Management page, depending on the configuration change.

Caution:

To help identify alarms raised during a configuration change, ensure that any existing system alarms are cleared before you begin.

On the **Hosts** tab of the **System Inventory** pane, the status **Config out-of-date** is shown for hosts affected by the change. Each host with this status must be locked and then unlocked to update its configuration and clear the alarm.

Changing the OAM IP Configuration

You can change the External OAM subnet, floating IP address, controller addresses, and default gateway at any time after installation.

During software installation, HP Helion OpenStack Carrier Grade is configured with an OAM network subnet and related IP addresses. You can change these addresses using the web administration interface or the CLI.

Caution:

Access to the OAM network is interrupted during this procedure. When a swact is performed on the controllers, the newly active controller uses the changed OAM IP addresses. The existing OAM IP addresses are no longer valid, and you must use the new OAM IP addresses to reconnect to the controller. Changes to external OAM access routing settings may also be required. In addition, VNC console access to compute-node hosts is interrupted until the hosts are locked and unlocked.

Before changing the OAM IP configuration, review the Fault Management page and ensure that any existing system alarms are cleared.

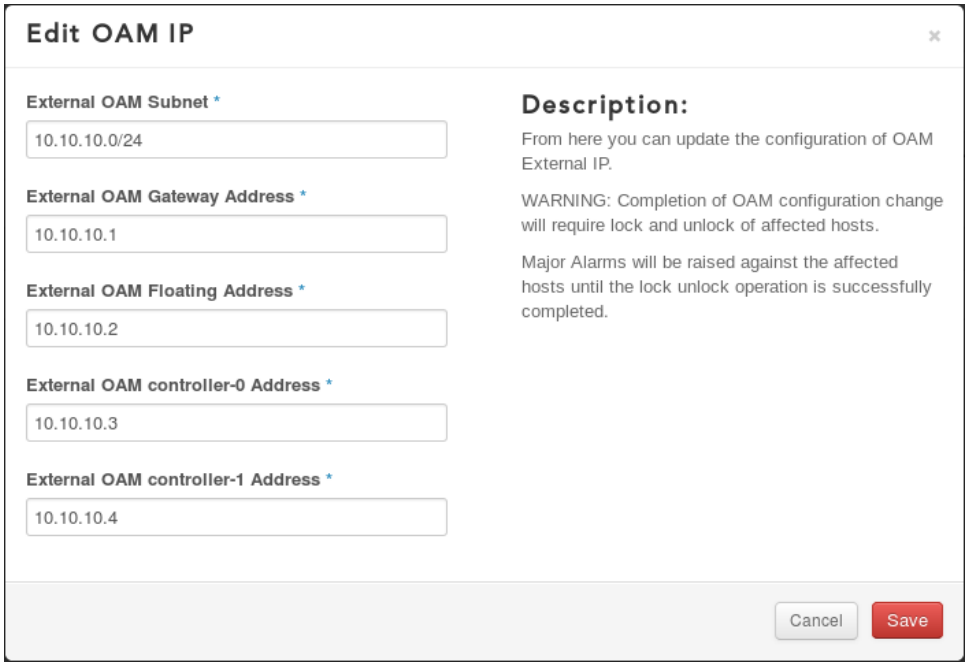
1. In the HP Helion OpenStack Carrier Grade web administration interface, open the **System Configuration** pane.

   The **System Configuration** pane is available from the **System Panel** section of the **Admin** menu.
2. Select the **OAM IP** tab.

   The **OAM IP** page appears, showing the currently defined OAM network configuration.

3. Click **Edit OAM IP**.

   The **Edit OAM IP** dialog box appears.



4. Replace the IP addresses with different ones as required.
5. Click **Save**.

   This raises major alarms against the controllers and services. You can view the alarms on the Fault Management page.
6. Lock and unlock the controllers to clear the alarms.

   Open the **Hosts** list by clicking **Inventory** on the **System Panel** section of the **Admin** menu, and then selecting the **Hosts** tab. Hosts requiring attention are shown with the status **Config out-of-date**.

   To lock or unlock a host, click **More** for the host and then use the menu selections.

   a) Lock the standby controller.

      Wait for the lock operation to be completed.
   b) Unlock the standby controller.

      Wait for the host to become available. It is updated with the new OAM configuration, and its error message is cleared.
   c) Perform a swact on the active controller.

      Click **More** > **Swact Host** > for the active controller.

      Note:

      After the swact, HP Helion OpenStack Carrier Grade begins using the new OAM network configuration. If you have changed the OAM subnet or the OAM floating IP address, the current web session is terminated. To continue using the web administration interface, you must start a new session and log in using the new IP address. You may also need to update router tables to establish a connection.
   d) Lock the original controller (now in standby mode).

      Wait for the lock operation to be completed.
   e) Unlock the original controller.

Wait for it to become available. It is updated with the new OAM configuration, and its error message is cleared.

7. Lock and unlock each compute node.

   Caution:

   Before locking a compute node, ensure that sufficient resources are available on other hosts to migrate any running instances.

8. Ensure that the **250.001 Configuration out-of-date** alarms are cleared for all hosts.

   If any alarms are present, clear them by locking and unlocking the affected host. In most cases, the alarms are cleared within one minute. If other system configuration operations are pending for the host, allow more time for the alarms to be cleared.

Changing the DNS Server Configuration

You can change the DNS servers defined for HP Helion OpenStack Carrier Grade at any time after installation.

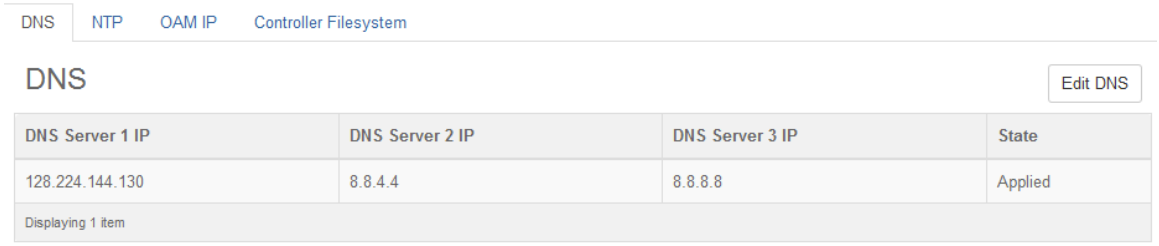
During software installation, HP Helion OpenStack Carrier Grade is configured with up to two DNS server IP addresses. You change these addresses using the web administration interface or the CLI.

1. In the HP Helion OpenStack Carrier Grade web administration interface, open the **System Configuration** pane.

   The **System Configuration** pane is available from the **System Panel** section of the **Admin** menu.

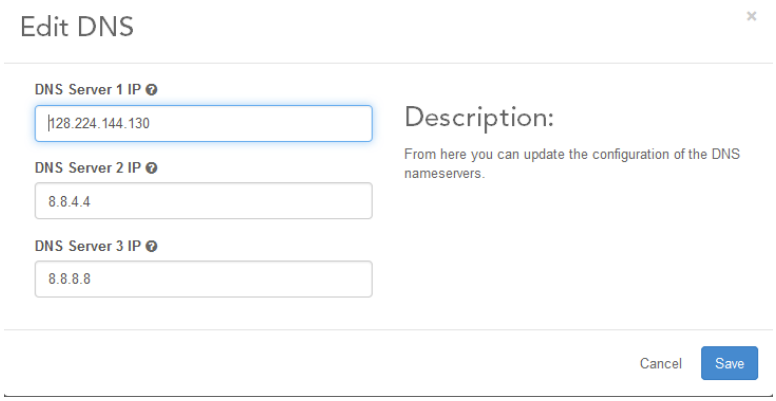
2. Select the **DNS** tab.

   The **DNS** page appears, showing the currently defined DNS servers and their IP addresses.



3. Click **Edit DNS**.

   The **Edit DNS** dialog box appears.



4. Replace the DNS Server IP addresses with different ones as required.

Upon completion of the DNS configuration change, a **250.001 Configuration out-of-date** alarm is raised temporarily, and then cleared automatically by the system.

Changing the NTP Server Configuration

You can change the NTP server addresses defined for HP Helion OpenStack Carrier Grade at any time after installation.

During software installation, HP Helion OpenStack Carrier Grade is configured with up to three NTP server IP addresses. You change these addresses using the web administration interface or the CLI.

Before changing NTP server addresses, review the Fault Management page and ensure that any existing system alarms are cleared.

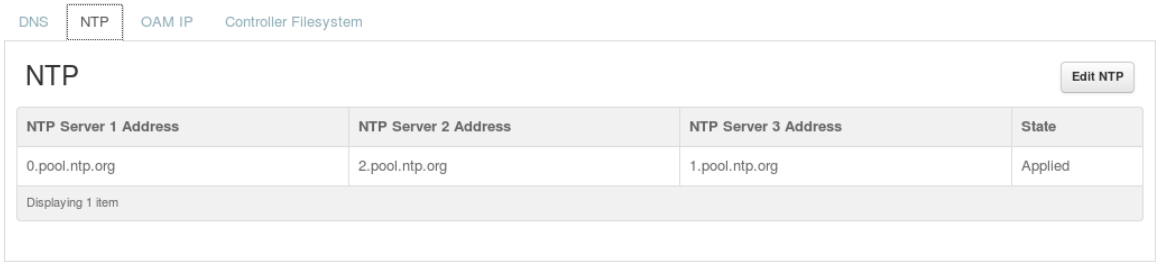
Caution:

For the HP Helion OpenStack Carrier Grade system to use FQDN servers instead of IPv4 servers, at least one valid DNS server must be specified.

1. In the HP Helion OpenStack Carrier Grade web administration interface, open the **System Configuration** pane.

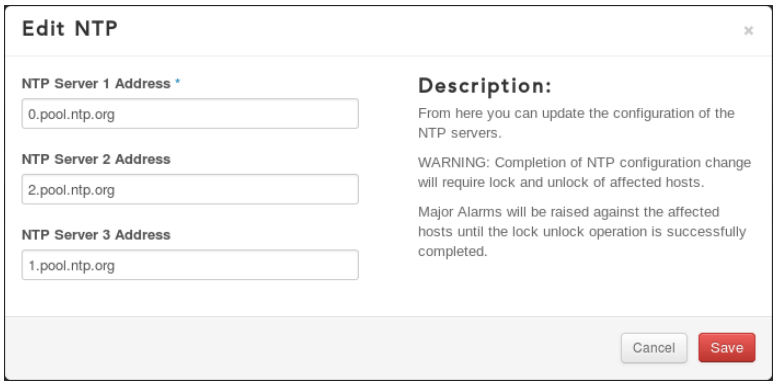
   The **System Configuration** pane is available from the **System Panel** section of the **Admin** menu.
2. Select the **NTP** tab.

   The **NTP** page appears, showing the currently defined NTP servers and their IP addresses.



3. Click **Edit NTP**.

   The **Edit NTP** dialog box appears.



4. Replace the NTP Server IP addresses or names with different ones as required.

   If you specify the NTP servers using FQDNs you must have at least one valid DNS server defined on your system. You must use IP addresses otherwise.
5. Click **Save**.

   This raises major alarms against the controllers and services. You can view the alarms on the Fault Management page.
6. Lock and unlock the controllers to clear the alarms.

   Open the **Hosts** list by clicking **Inventory** on the **System Panel** section of the **Admin** menu, and then selecting the **Hosts** tab. Hosts requiring attention are shown with the status **Config out-of-date**.

To lock or unlock a host, click **More** for the host and then use the menu selections.

a) Lock the standby controller.

   Wait for the lock operation to be completed.

b) Unlock the standby controller.

   Wait for the host to become available. It is updated with the new NTP server addresses, and its error message is cleared.

c) Perform a swact on the active controller.

   Click **More** > **Swact Host** > for the active controller.

d) Lock the original controller (now in standby mode).

   Wait for the lock operation to be completed.

e) Unlock the original controller.

   Wait for it to become available. It is updated with the new NTP server addresses, and its error message is cleared.

7. Ensure that the **250.001 Configuration out-of-date** alarms are cleared for both controllers.

Changing Storage Space Allotments on the Controller

You can change the allotments for controller-based storage at any time after installation. This includes the space allotted for database, image, and backup storage, and for clusters using an LVM backend, the space allotted for Cinder volume storage.

Note:

Cinder volume storage is allotted using 3PAR, not on the controller node..

During software installation, HP Helion OpenStack Carrier Grade is configured with the following storage allotments.

Database storage

The storage allotment for the OpenStack database.

Image storage

The size of the partition to use for image storage.

Backup storage

The storage allotment for backup operations.

Volume storage

For a system using LVM storage, the storage allotment for all Cinder volumes used by guest instances.

You can change these allotments using the **System Configuration** controls in the web administration interface, or using the CLI. For CLI instructions, see *System Configuration Management Using the CLI* on page 88.

To accommodate the changes, there must be enough disk space on the controller, including headroom needed to complete the operation. The headroom required is 45 GiB on the primary disk for a cluster using an LVM backend, or 65 GiB for a cluster using a 3PAR backend. This is in addition to the space required for the new allotments. The requested changes are checked against available space on the affected disks; if there is not enough, the changes are disallowed.

To provide more space, you can replace the affected disk or disks. Database, image, and backup storage use space on the primary disk. Cinder volume storage (on a cluster with an LVM backend) uses space on a disk selected by device node number during controller configuration. The replacement disk must occupy the same device node number. Changes to the Cinder volume storage may also affect the primary disk because of the headroom requirement.

To pass the disk-space checks, any replacement disks must be installed before the allotments are changed.

Caution:

The configurations for **controller-0** and **controller-1** must be identical. If you make changes to one, you must make the same changes to the other.

Changes to the storage allotments require a reinstallation of the HP Helion OpenStack Carrier Grade host software on the controllers, even if the primary disk is not replaced.

Note:

Performing a host reinstall successfully is dependent on your BIOS boot order. Observe the following tips.

- Prior to installation, the BIOS should be configured to allow booting from disk and the network.
- During the host re-installation, it is advisable to have a console serial cable attached to the host to observe the boot progress.
- BIOS boot order should be:
  1. The boot partition.

     Typically, this is the disk associated with `/dev/sda` and is as defined in `/proc/cmdline` when the load is booted.
  2. The NIC on the boot interface (e.g. management or pxe network).
- Set BIOS boot options to ensure failsafe boot if available, e.g. rotating through available boot interfaces, watchdog timer on boot, retry boot interfaces etc.

If the BIOS boot still fails to progress, it may be necessary to force boot via the network interfaces via the BIOS boot option.

Before changing storage allotments, prepare as follows:

- Calculate your system storage requirements. Include the headroom required for changes to the storage space allotments.
- Record the current configuration settings in case they need to be restored (for example, because of an unexpected interruption during changes to the system configuration). Consult the configuration plan for your system.

1. If necessary, install replacement disks in the controllers.

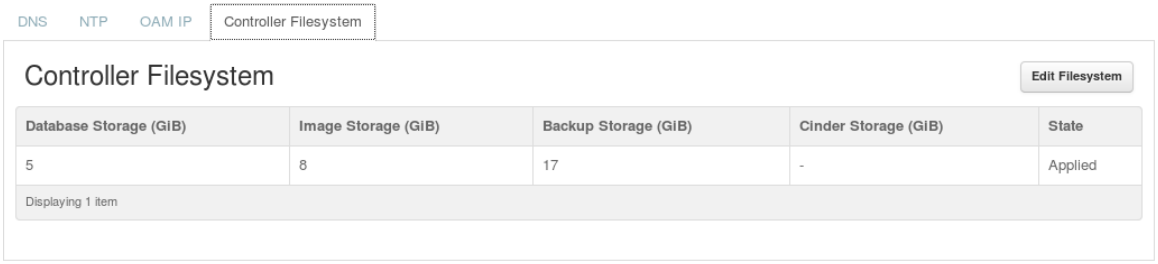
   If you do not need to replace disks, you can skip this step. To determine whether you need to replace disks, calculate your storage requirements. Be sure to include the headroom required on the primary disk.

   To replace disks in the controllers, follow these steps.

   a) Lock the standby controller.
   b) Power down the standby controller.
   c) Make any required hardware changes.
   d) Power up the standby controller.
   e) Reinstall the HP Helion OpenStack Carrier Grade software on the standby controller.
   f) Unlock the standby controller.
   g) Swact the controllers.
   h) Repeat these steps for the new standby controller.
2. Edit the disk storage allotments.
   a) In the HP Helion OpenStack Carrier Grade web administration interface, open the **System Configuration** pane.

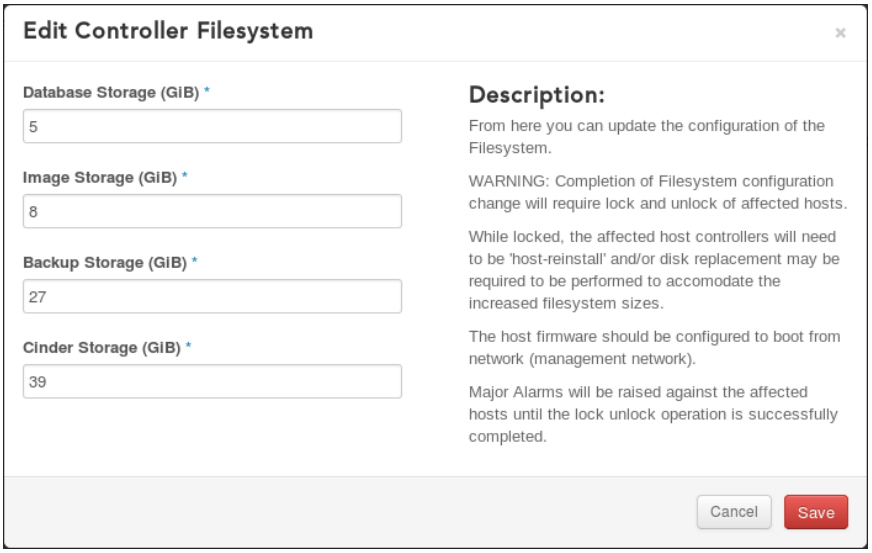
      The **System Configuration** pane is available from the **System Panel** section of the **Admin** menu.
   b) Select the **Controller Filesystem** tab.

      The **Controller Filesystem** page appears, showing the currently defined storage allotments.



c) Click **Edit Filesystem**.

The **Edit Controller Filesystem** dialog box appears.



Note:

The **Cinder storage (GiB)** field is present only for clusters using the LVM backend.

d) Replace the storage allotments as required.

e) Click **Save**.

This raises major alarms against the controllers (**250.001 Configuration out-of-date**). You can view the alarms on the Fault Management page. In addition, the status **Config out-of-date** is shown for the controllers in the **Hosts** list.

3. Lock the standby controller.
4. Reinstall the HP Helion OpenStack Carrier Grade software on the standby controller.

   This step is required to update the system configuration, even if you have already reinstalled the HP Helion OpenStack Carrier Grade software as part of the disk replacement procedure.
5. Unlock the standby controller.

   When the controller is unlocked, the **250.001 Configuration out-of-date** alarms against the controller are cleared.
6. Swact the controllers.
7. Lock the new standby controller.
8. Reinstall the HP Helion OpenStack Carrier Grade software on the controller.
9. Unlock the controller.
10. Confirm that the **250.001 Configuration out-of-date** alarms are cleared for both controllers.

After making these changes, ensure that the configuration plan for your system is updated with the new storage allotments and disk sizes.

System Configuration Management Using the CLI

You can use the CLI to make changes to the system configuration after installation.

For information about the types of configuration changes you can make, and guidelines for implementing them on an operational system, see *System Configuration Management* on page 79. During some configuration changes, system alarms are raised; for help viewing them from the CLI, see *System Alarms CLI Commands* on page 176.

To make configuration changes using the CLI, you must log in as user **wrsroot** to the active controller and source the script `/etc/nova/openrc` to obtain Keystone administrative privileges. See *Linux User Accounts* on page 18 for details.

With the exception of the DNS Server Configuration, the configuration changes described in this section require that you lock and unlock the indicated hosts after the configuration change is made. For more information about locking and unlocking hosts, see *Host Inventory* on page 103.

OAM IP Configuration

To view the existing OAM IP configuration, use the following command.

```
~(keystone_admin)$ system oam-show
+-----------------+--------------------------------------+
| Property        | Value                                |
+-----------------+--------------------------------------+
| uuid            | bacf5c9d-47a1-43df-845a-f38a88e1dcd4 |
| istate          | applied                              |
| oam_lag_ifc     | False                                |
| oam_ifcs        | eth0                                 |
| oam_mtu         | 1500                                 |
| oam_subnet      | 10.10.10.0/24                        |
| oam_gateway_ip  | 10.10.10.1                           |
| oam_floating_ip | 10.10.10.2                           |
| oam_c0_ip       | 10.10.10.3                           |
| oam_c1_ip       | 10.10.10.4                           |
| isystem_uuid    | 1bbd29be-c6a2-4e9b-93ce-a8dac3ce8e5c |
| recordtype      | reconfig                             |
| created_at      | 2014-12-23T21:15:34.097509+00:00     |
| updated_at      | 2014-12-31T19:41:43.787097+00:00     |
+-----------------+--------------------------------------+
```

To change the OAM IP subnet, floating IP address, gateway IP address, or controller IP addresses, use the following command syntax.

```
~(keystone_admin)$ system oam-modify oam_subnet=subnet/netmask \
oam_gateway_ip=gateway_ip_address \
oam_floating_ip=floating_IP_address \
oam_c0_ip=controller-0_IP_address \
oam_c1_ip=controller-1_ip_address \
action=apply
```

For example:

```
~(keystone_admin)$ system oam-modify oam_subnet=10.10.10.0/24 \
oam_gateway_ip=10.10.10.1 \
oam_floating_ip=10.10.10.2 \
oam_c0_ip=10.10.10.3 \
oam_c1_ip=10.10.10.4 \
action=apply
```

After changing the OAM server configuration, you must lock and unlock the controllers. This process requires a swact on the controllers. Then you must lock and unlock the compute nodes one at a time, ensuring that sufficient resources are available to migrate any running instances.

DNS Server Configuration

To view the existing DNS server configuration, use the following command.

```
~(keystone_admin)$ system dns-show
+--------------+--------------------------------------+
| Property     | Value                                |
+--------------+--------------------------------------+
| uuid         | cd2d645f-d8ca-4344-995b-aab480b1951d |
| recordtype   | reconfig                             |
| istate       | applied                              |
| nameservers  | 8.8.8.8,8.8.4.4                      |
| isystem_uuid | 1bbd29be-c6a2-4e9b-93ce-a8dac3ce8e5c |
| created_at   | 2014-12-23T21:15:34.127239+00:00     |
| updated_at   | 2014-12-31T19:41:43.806842+00:00     |
+--------------+--------------------------------------+
```

To change the DNS server IP addresses, use the following command syntax. The nameservers option takes a comma-delimited list of DNS server IP addresses.

```
~(keystone_admin)$ system dns-modify \
nameservers=IP_address_1[,IP_address_2][,IP_address_3] action=apply
```

For example:

```
~(keystone_admin)$ system dns-modify \
nameservers=8.8.8.8,8.8.4.4 action=apply
```

NTP Server Configuration

To view the existing NTP server configuration, use the following command.

```
~(keystone_admin)$ system ntp-show
+--------------+------------------------------------------------+
| Property     | Value                                          |
+--------------+------------------------------------------------+
| uuid         | 33bf8ad6-9426-4e10-b5b2-0be1b23a22c9           |
| recordtype   | reconfig                                       |
| istate       | applied                                        |
| ntpservers   | 0.pool.ntp.org,2.pool.ntp.org,1.pool.ntp.org   |
| isystem_uuid | 1bbd29be-c6a2-4e9b-93ce-a8dac3ce8e5c           |
| created_at   | 2014-12-23T21:15:34.145488+00:00               |
| updated_at   | 2014-12-31T19:53:44.343981+00:00               |
+--------------+------------------------------------------------+
```

To change the NTP server IP addresses, use the following command syntax. The ntpservers option takes a comma-delimited list of NTP server names.

```
~(keystone_admin)$ system ntp-modify \
ntpservers=server_1[,server_2][,server_3] action=apply
```

For example:

```
~(keystone_admin)$ system ntp-modify \
ntpservers=0.north-america.pool.ntp.org,\
0.north-america.pool.ntp.org,0.north-america.pool.ntp.org \
```

```
action=apply
```

After changing the NTP server configuration, you must lock and unlock both controllers. This process requires a swact on the controllers.

Controller Storage Configuration

To view the existing storage configuration, use the following command.

```
~(keystone_admin)$ system controllerfs-show
+--------------+----------------------------------------+
| Property     | Value                                  |
+--------------+----------------------------------------+
| uuid         | b1242c7f-7cb1-4878-a088-aac02c3f424b |
| recordtype   | reconfig                               |
| istate       | applied                                |
| database_gib | 5                                      |
| image_gib    | 8                                      |
| backup_gib   | 27                                     |
| cinder_gib   | 39                                     |
| isystem_uuid | 1bbd29be-c6a2-4e9b-93ce-a8dac3ce8e5c |
| created_at   | 2014-12-23T21:15:33.970585+00:00     |
| updated_at   | 2014-12-31T19:54:44.321612+00:00     |
+--------------+----------------------------------------+
```

Before you can change storage allotments, you must have enough disk space, including additional headroom required to complete the operation. If there is not enough disk space, changes to the storage allotment are disallowed. For more information, see *Changing Storage Space Allotments on the Controller* on page 85.

You can replace the disks in the controllers to provide more space. For more information, see *Managing Controller Nodes* on page 113. To support storage changes, you must reinstall the HP Helion OpenStack Carrier Grade software on each controller during the disk replacement procedure. This is required even if the primary disk is not directly affected by the change. To reinstall the software, you can use the following command:

```
~(keystone_admin)$ system host-reinstall controller_name
```

Note:

Performing a host reinstall successfully is dependent on your BIOS boot order. Observe the following tips.

- Prior to installation, the BIOS should be configured to allow booting from disk and the network.
- During the host re-installation, it is advisable to have a console serial cable attached to the host to observe the boot progress.
- BIOS boot order should be:
  1. The boot partition.

     Typically, this is the disk associated with `/dev/sda` and is as defined in `/proc/cmdline` when the load is booted.
  2. The NIC on the boot interface (e.g. management or pxe network).
- Set BIOS boot options to ensure failsafe boot if available, e.g. rotating through available boot interfaces, watchdog timer on boot, retry boot interfaces etc.

If the BIOS boot still fails to progress, it may be necessary to force boot via the network interfaces via the BIOS boot option.

To change the database, image, or backup storage allotment, or the Cinder volume storage allotment on a system with an LVM storage backend, use the following command syntax, where the allotments are in GiB.

```
~(keystone_admin)$ system controllerfs-modify \
database_gib=database_allotment \
image_gib=image_allotment \
```

```
backup_gib=backup_allotment \
cinder_gib=cinder_volume_allotment \
action=apply
```

For example:

```
~(keystone_admin)$ system controllerfs-modify \
database_gib=10 image_gib=13 backup_gib=22 action=apply
```

After changing the controller storage configuration, you must lock each controller and reinstall the HP Helion OpenStack Carrier Grade software, even if you have already done so as part of changing the controller disks. You can use the following command:

```
~(keystone_admin)]$ system host-reinstall controller_name
```

Then you can unlock the controller to clear any **Configuration out-of-date** alarms. A controller swact is required to update both controllers.

Adding an Infrastructure Network Using the CLI

If an infrastructure network is not installed during initial configuration, you can add one later using the CLI.

Note:

On a cluster with an infrastructure network, each host must have an infrastructure interface before it can be unlocked. Ensure that all hosts have the required hardware.

For a system with an LVM storage backend, the infrastructure network is optional. It can be used to offload the internal management network on clusters with many compute nodes. For more information, see *Network Planning* on page 36.

You can view the existing infrastructure network configuration using the following command on the active controller (assumed to be **controller-0** for this discussion).

```
~(keystone_admin)$ system infra-show
```

```
+--------------+--------------------------------------+
| Property     | Value                                |
+--------------+--------------------------------------+
| uuid         | cc549ef3-5c81-4fbf-8c50-5163c89e5039 |
| istate       | applied                              |
| infra_subnet | None                                 |
| infra_start  | None                                 |
| infra_end    | None                                 |
| isystem_uuid | 1bbd29be-c6a2-4e9b-93ce-a8dac3ce8e5c |
| recordtype   | reconfig                             |
| created_at   | 2014-12-23T21:15:34.074881+00:00     |
| updated_at   | 2014-12-31T19:38:43.603320+00:00     |
+--------------+--------------------------------------+
```

You can add an infrastructure network after initial installation. For this operation, all nodes except the active controller must be locked. During the change, the nodes cannot be unlocked until the new configuration is applied to both controllers. In addition, before a node can be unlocked, an infrastructure interface must be configured on the host.

1. Lock all hosts except the active controller.

   For each host, use the following command.

   ```
   ~(keystone_admin)$ system host-lock hostname
   ```

2. Add an infrastructure interface on the standby controller.

   Note:

   You can also add an infrastructure network to a shared interface as a VLAN-tagged network. For more information, see *Shared (VLAN) Ethernet Interfaces* on page 24.

   To add an infrastructure network to a dedicated interface, use the following command, specifying a name for the network and the port to use for the network connection. To identify the port to use for an infrastructure network, consult your configuration plan.

```
~(keystone_admin)$ system host-if-modify -nt infra \
-n interfacename hostname port
```

   For example:

```
~(keystone_admin)$ system host-if-modify -nt infra \
-n infra0 controller-1 eth3
+------------------+--------------------------------------+
| Property         | Value                                |
+------------------+--------------------------------------+
| ifname           | infra0                               |
| networktype      | infra                                |
| iftype           | ethernet                             |
| ports            | [u'eth3']                            |
| providernetworks | None                                 |
| imac             | 08:00:27:7a:fa:a5                    |
| imtu             | 1500                                 |
| aemode           | active_standby                       |
| schedpolicy      | None                                 |
| txhashpolicy     | layer2                               |
| uuid             | d3efde57-c475-402e-acad-34f5029f3988 |
| ihost_uuid       | 62657285-cc9c-45a2-962e-ecb01647916f |
| vlan_id          | None                                 |
| uses             | []                                   |
| used_by          | []                                   |
| created_at       | 2014-12-31T20:13:16.326992+00:00     |
| updated_at       | None                                 |
+------------------+--------------------------------------+
```

3. Add the same infrastructure network on the active controller.

   For example:

```
~(keystone_admin)$ system host-if-modify -nt infra \
-n infra0 controller-0 eth3
+------------------+--------------------------------------+
| Property         | Value                                |
+------------------+--------------------------------------+
| ifname           | infra0                               |
| networktype      | infra                                |
| iftype           | ethernet                             |
| ports            | [u'eth3']                            |
| providernetworks | None                                 |
| imac             | 08:00:27:51:58:6b                    |
| imtu             | 1500                                 |
| aemode           | active_standby                       |
| schedpolicy      | None                                 |
| txhashpolicy     | layer2                               |
| uuid             | 66c4543e-6bfd-4ede-9b8a-4d091cf290a9 |
| ihost_uuid       | 3c11607b-1550-4e99-a46f-2580c18683da |
| vlan_id          | None                                 |
| uses             | []                                   |
| used_by          | []                                   |
```

```
| created_at       | 2014-12-31T20:29:51.759507+00:00     |
| updated_at       | None                                 |
+------------------+--------------------------------------+
```

4. Specify the infrastructure subnet.

```
~(keystone_admin)$ system infra-modify \
infra_subnet=subnet/mask action=apply
```

For example:

```
~(keystone_admin)$ system infra-modify \
infra_subnet=192.168.205.0/24 action=apply

+--------------+--------------------------------------+
| Property     | Value                                |
+--------------+--------------------------------------+
| uuid         | cc549ef3-5c81-4fbf-8c50-5163c89e5039 |
| istate       | applying                             |
| infra_subnet | 192.168.205.0/24                     |
| infra_start  | None                                 |
| infra_end    | None                                 |
| isystem_uuid | 1bbd29be-c6a2-4e9b-93ce-a8dac3ce8e5c |
| recordtype   | reconfig                             |
| created_at   | 2014-12-23T21:15:34.074881+00:00     |
| updated_at   | 2015-01-02T16:06:39.275800+00:00     |
+--------------+--------------------------------------+
```

5. Reboot the standby controller.

```
~(keystone_admin)$ system host-reboot controller-1
```

This updates its configuration.

Wait for the standby controller to reboot. To monitor its status, use the following command.

```
~(keystone_admin)$ watch -n 5 system host-list
```

This displays a refreshed host list every five seconds. Monitor the output until the standby controller is shown as **locked** and **online**. To stop monitoring, enter `CTRL-C`.

Caution:

To prevent potential service conflicts due to inconsistent controller network configurations, do not unlock the standby controller until the active controller is also rebooted.

6. Reboot the active controller.

```
~(keystone_admin)$ sudo reboot
The system is going down for reboot NOW!
```

Wait for the controller to reboot.

7. Log in to the active controller and become the Keystone **admin** user.

```
$ source /etc/nova/openrc
```

8. Unlock the standby controller.

```
~(keystone_admin)$ system host-unlock controller-1
+-----------------+--------------------------------------+
| Property        | Value                                |
+-----------------+--------------------------------------+
```

```
| action          | none                                 |
| administrative  | locked                               |
| availability    | online                               |
| bm_ip           |                                      |
| bm_mac          |                                      |
| bm_type         | None                                 |
| bm_username     |                                      |
| capabilities    | {}                                   |
| created_at      | 2014-12-24T16:44:08.749540+00:00     |
| cstatus         |                                      |
| hostname        | controller-1                         |
| iconfig_applied | 76aa6a81-620a-4ba2-85c3-113d1c42f5ec |
| iconfig_fini    | 76aa6a81-620a-4ba2-85c3-113d1c42f5ec |
| iconfig_target  | 76aa6a81-620a-4ba2-85c3-113d1c42f5ec |
| id              | 5                                    |
| invprovision    | unprovisioned                        |
| location        | {u'locn': u'Rack 1 Pos 2'}           |
| mgmt_ip         | 192.168.204.4                        |
| mgmt_mac        | 08:00:27:ba:e8:52                    |
| operational     | disabled                             |
| personality     | controller                           |
| reserved        | False                                |
| serialid        | None                                 |
| task            | Unlocking                            |
| updated_at      | 2015-01-02T17:12:56.464592+00:00     |
| uptime          | 1955                                 |
| uuid            | 62657285-cc9c-45a2-962e-ecb01647916f |
+-----------------+--------------------------------------+
```

9. Add infrastructure interfaces to each compute node.

For each node, use the following command.

```
~(keystone_admin)$ system host-if-modify -nt infra \
-n networkname hostname port
+------------------+--------------------------------------+
| Property         | Value                                |
+------------------+--------------------------------------+
| ifname           | infra0                               |
| networktype      | infra                                |
| iftype           | ethernet                             |
| ports            | [u'eth0']                            |
| providernetworks | None                                 |
| imac             | 08:00:27:20:d2:d1                    |
| imtu             | 1500                                 |
| aemode           | active_standby                       |
| schedpolicy      | None                                 |
| txhashpolicy     | layer2                               |
| uuid             | d664ad66-58fa-4378-b906-ae417d58dbf0 |
| ihost_uuid       | 43b462ac-05fc-43dd-82de-160c7ab0923f |
| vlan_id          | None                                 |
| uses             | []                                   |
| used_by          | []                                   |
| created_at       | 2015-01-02T17:33:37.138694+00:00     |
| updated_at       | None                                 |
+------------------+--------------------------------------+
```

You can now unlock the compute nodes. This clears any **Configuration out-of-date** errors.

Chapter 4

Managing Hardware Resources

Topics:

HP Helion OpenStack Carrier Grade reports status information about the hosts in the cluster, and provides a set of tools to operate on them to execute commands, configure parameters, and monitor status.

Cluster Overview

The **Overview** page appears when you log into the HP Helion OpenStack Carrier Grade web management interface as the administrator. It presents a dashboard providing a quick health-check overview of the cluster.

The **Overview** page is an enhanced version of its OpenStack counterpart. It includes the following features:

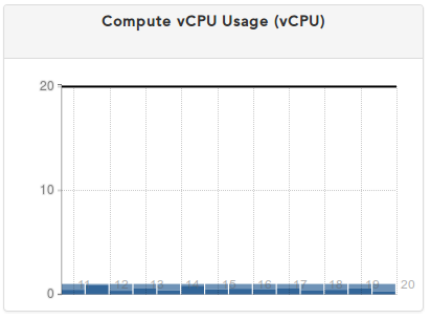
- dynamically updated, relocatable resource charts. Click the top of any chart, and drag it to the desired location to customize the presentation as desired. The layout changes are persistent across logins.
- dynamically updated, relocatable Host Server Status summary table. Presents the total number of hosts currently in one of the following states:
  - locked
  - unlocked
  - available
  - unavailable
  - degraded
- dynamically updated, relocatable Alarms summary table. Lists the current count of critical, major, minor, and warning alarms.

Common attributes of the resource pages are:

- All resource charts display resource information in real-time, spanning over a time-window of ten minutes.
- Labels along the X-axis present the number of minutes elapsed since the last full-time hour of the system clock.
- Hovering the mouse over the charts overlays information windows with detailed values.
- History information is lost when the **Overview** page is no longer displayed.

The following charts are available:

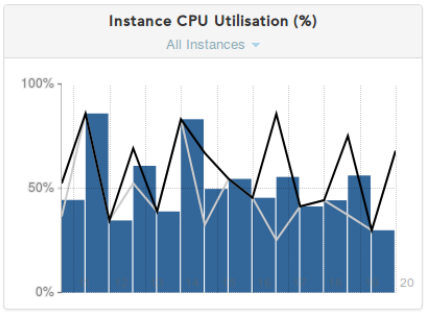
Compute vCPU Usage



Presents information about the number of virtual CPUs across the entire cluster, available and allocated to virtual machines, and their average level of utilization. Elements in this chart are as follows:

- Y-axis is in units vCPUs.
- A solid black line tracking the number of available vCPUs. This line settles at a constant value at the top of the chart, unless the number of compute nodes changes over time.
- Light blue bars indicating the number of allocated vCPUs, that is, the number of vCPUs in use by virtual machines (one in the example).
- Dark blue bars indicating the average level of utilization of the allocated vCPUs (varies from 30 to 83% in the example), presented as a relative height with respect to the light blue bars.

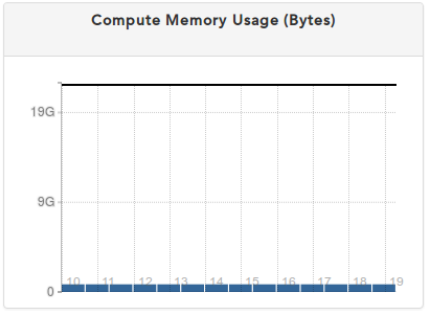
Instance CPU Utilization



Presents information about virtual CPU utilization levels on virtual machines, across the entire cluster or for individual instances. Elements in this chart are as follows:

- Y-axis displays the virtual CPU utilization level, from 0 to 100%.
- A solid black line tracking the maximum utilization level.
- Dark blue bars indicating the average utilization level.
- A gray line tracking the minimum utilization level.
- A filter drop-down menu to select the virtual machines on which to report utilization levels. The default value for this filter is "All Instances," and therefore it reports the same utilization levels indicated by the dark blue bars in the Compute vCPU Usage chart.

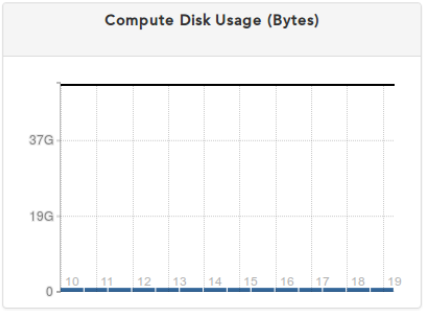
Compute Memory Usage



Presents information about memory usage levels on the compute nodes across the entire cluster. Elements in this chart are as follows:

- Y-axis is in units of bytes.
- A solid black line tracking the available aggregated compute memory. This line usually settles as a constant value at the top of the chart, unless the number of compute nodes changes over time.
- Dark blue bars indicating the amount of memory allocated to virtual machines (less than 1 GB in the example).

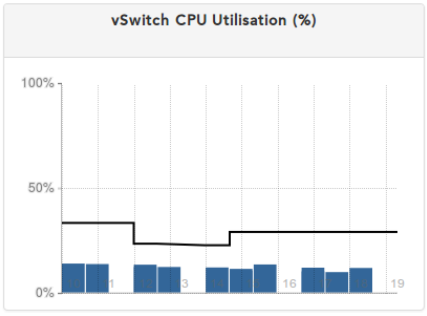
Compute Disk Usage



Presents information about disk usage levels on the compute nodes across the entire cluster. Elements in this chart are as follows:

- Y-axis is in units of bytes.
- A solid black line tracking the available aggregated storage capacity. This line usually settles as a constant value at the top of the chart.
- Dark blue bars indicating the amount of disk space allocated to virtual machines (about 1 GB in the example).

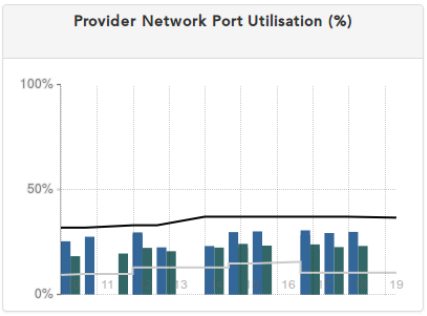
AVS CPU Utilization



Presents information about AVS CPU utilization levels on the compute nodes across the entire cluster. Elements in this chart are as follows:

- Y-axis displays the AVS CPU utilization level, from 0 to 100%.
- A solid black line tracking the maximum utilization level.
- Dark blue bars indicating the average utilization level.

Provider Network Port Utilization



Presents information about network traffic levels on provider networks across the entire cluster. Elements in this chart are as follows:

- Y-axis displays the port utilization level, from 0 to 100%.
- A solid black line tracking the maximum utilization level.
- Pairs of dark blue and dark green bars indicating the average utilization level, egress and ingress (Tx and Rx) traffic respectively.
- A gray line tracking the minimum utilization level.

Resource Usage

Usage of system resources is monitored by Ceilometer, the standard OpenStack mechanism for collecting and distributing performance monitoring samples from the cluster. The HP Helion OpenStack Carrier Grade cluster extends Ceilometer with improved reports and new tools to facilitate offline analysis of the collected data.

Performance Monitor (PM) samples are periodically collected from different resources such as hosts, virtual machine instances, AVS, and others. They include CPU and memory utilization, network traffic counters, storage space, and several more. By default the samples are stored in a database which is used for reporting activities such as:

- command line queries and graphical charts
- triggering of threshold alarms
- triggering of threshold actions. The most common example is Heat actions to scale up and down a resource.

The Ceilometer database is periodically cleaned in order to contain its size and prevent the overuse of storage resources on the controller nodes.

The Ceilometer implementation in HP Helion OpenStack Carrier Grade improves on several aspects. They are described in the following sections.

Optimized Ceilometer Filters

The Ceilometer implementation in HP Helion OpenStack Carrier Grade is specially tuned to allow time-sensitive actions, such as Heat autoscaling, to take place in response to relatively high-frequency sample rates of system events. The size of both the Ceilometer database and the CSV files are maintained under control at all times, while still enabling the required system actions to take place in real-time.

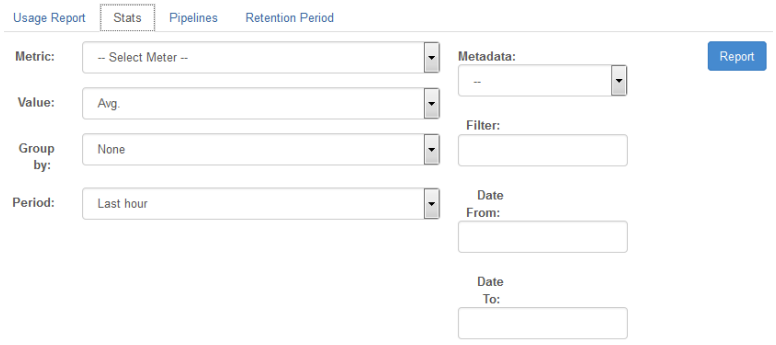
On-Line Ceilometer Reports

On-line, on-demand, Ceilometer reports in HP Helion OpenStack Carrier Grade benefit from several improvements over the stock OpenStack reporting tool, including:

- optimized Ceilometer database queries
- improved naming of menu entries on pull-down menus
- use of brief meter descriptions and human-readable legends on the charts

Additionally, all database queries are user-initiated, as opposed to event-initiated. This provides a different user workflow, whereby all required report parameters and filters are configured first, before the database query is executed

Access to Ceilometer reports is available from the **Stats** tab on the **Resource Usage Overview** page. This page can be accessed from the web management interface by selecting the option Resource Usage on the **Admin** tab of the system panel. The **Stats** tab is presented as follows:



The following fields are available:

Metric

The Ceilometer metric you want to query. Metrics are grouped by service, as follows:

- Compute (Nova)
- Network (Neutron)
- Image (Glance)
- Volume (Cinder)
- Switching (AVS)

On the CLI, use the following command to list the available metrics, and find their resource IDs:

```
~(keystone_admin)$ ceilometer meter-list
```

Value

The particular statistic you want to visualize. Select among Average, Minimum, Maximum, and Summation.

Group by

How to group the displayed charts, as follows:

None

The selected statistic is presented as a single line chart reporting the aggregate over all resources and projects (tenants).

Projects (Tenants)

The selected statistic is presented as a multiple line charts over all resources, one line per tenant.

Resource

The selected statistic is presented as a multiple line charts over all resources, one line per resource (hosts, instances, and so on).

Period

The period of time to be covered by the report. They include last 15 minutes, last 30 minutes, last hour, last day, last week, last 15 days, last 30 days, last year, and other. When selecting other, two new fields become available, Date From and Date To, allowing you to specify a specific time period.

Metadata

Metadata represents additional attributes collected with a metric. In this menu, you can select the specific attribute you want the report on.

The pull-down menu is auto-populated with the corresponding attributes for the selected metric. On the CLI, use the following command to list the metadata associated with a metric identified with its resource ID:

```
~(keystone_admin)$ ceilometer resource-show resource_id
```

Filter

Use this field to limit the report to show metric samples whose metadata attribute equals the specified value. The filter field is applied only when a specific metadata attribute is selected.

Once the selections are in place, click the Report button to display the report.

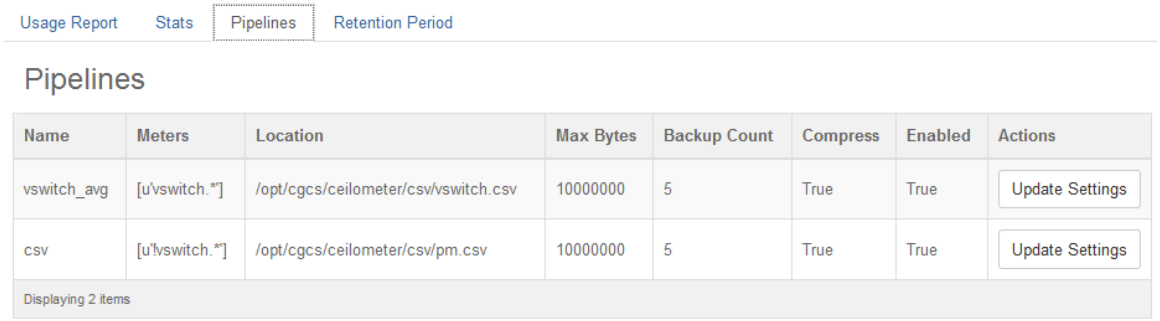
CSV Performance Monitoring Backend

HP Helion OpenStack Carrier Grade provides access to performance measurement samples in the form of comma-separated values (CSV) files. They provide the following benefits:

- off-line permanent storage of large samples history
- enables the use of off-line tools for data analysis, accounting, and archiving

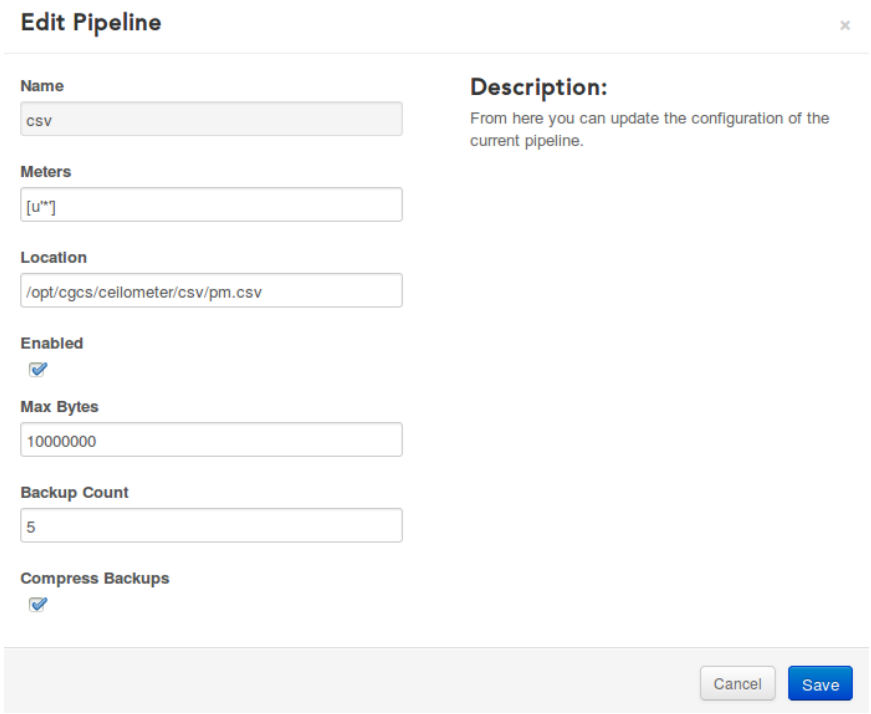
The CSV files are expected to be retrieved from the controllers using any suitable file transfer client application such as SFTP or SCP. This can be done manually by the system administrator, or automatically by an operations and support server (OSS/BSS) configured with the appropriate login credentials.

Recording of samples and management of CSV files is done through special Ceilometer pipelines available from the **Pipelines** tab, as illustrated below:



The **csv** pipeline collects system information with the exception of AVS switching meters. The latter are collected by the **vswitch_avg** pipeline.

Click the button **Update Settings** to configure the behavior of a pipeline. The **Edit Pipeline** window is presented as follows:



The following fields are available:

Name

A read-only field displaying the name of the pipeline

Meters

An editable field displaying the list of comma-separated meters included in the pipeline. The list must be enclosed in square brackets, as follows:

```
[meter1,meter2,...]
```

The following syntax rules apply to the specified meters:

- A meter specification is a text string of the form *metergroup.metersubgroup.meter*, for example, `disk.read.bytes`
- A meter specification supports a trailing wild-card to include all *meters* within a *metergroup*. For example, the text `disk.*` matches the meters `disk.read.bytes` and `disk.write.bytes`.

- A meter specification supports a trailing wild-card to include all *metersubgroups* within a *metergroup*. For example, the text `disk.read.*` matches the meter `disk.read.bytes`.
- Meter specifications support a leading exclamation mark to exclude the specified meter, as follows:

  ```
  [!meter1,!meter2,...]
  ```

  Such a pipeline includes all meters but the ones in the list.

  Exclamation marks cannot be applied selectively in a list of meters. Either all meters use them or none at all. The following list is therefore invalid:

  ```
  [meter1,!meter2,meter3]
  ```

Location

The absolute path to the CSV file on the controller.

You should use a path below `/opt/cgcs`. Otherwise, if the active controller fails, the `csv` files will not be available on the backup controller

Enabled

A check box used to enable or disable the pipeline.

Max Bytes

The maximum size, in bytes, of a CSV file. The default value is 10 MB.

Backup Count

The number of concurrent backup CSV files to maintain in a rotation ring, including the currently active file. When a CSV file reaches its maximum size, it is renamed as a backup file, and a new CSV file is opened for writing. Backup files older than the size of the rotation ring are automatically removed.

Compress Backups

A check box used to select whether or not to compress backup files in the rotation ring.

Retention Period

Performance samples are kept in the database for a limited period of time known as the *retention period*. Its default value is 86400 seconds, or 24 hours.

Click the **Edit Retention Period** button in the **Retention Period** tab to modify the current value. You must ensure that the configured value is equal or higher than the period over which you intend to gather statistics.

You can also control the retention period from the CLI. To view the current settings, use the following command.

```
~(keystone_admin)$ system pm-show
```

To change the retention period from the CLI, use the following command syntax.

```
~(keystone_admin)$ system pm-modify retention_secs=retention_period
```

For example:

```
~(keystone_admin)$ system pm-modify retention_secs=172800
```

Note:

Changes to the retention period cause **250.001 Configuration out-of-date** alarms to be raised briefly for the controller nodes. During this period, the status **Config out-of-date** is displayed for the controller nodes on the **Host** tab of the **Inventory** pane. These alarms are resolved and cleared automatically after a few seconds.

Managing Hardware Inventory

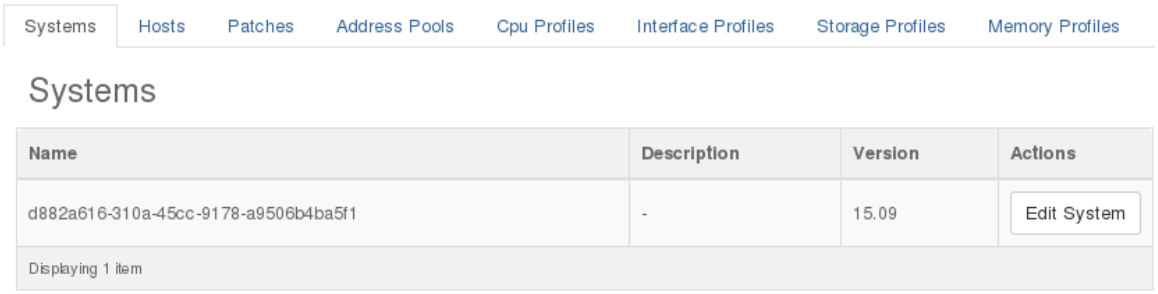
The maintenance and inventory component of HP Helion OpenStack Carrier Grade allows the administrator to install, configure, and manage the hosts of the HP Helion OpenStack Carrier Grade cluster.

To access the **System Inventory** page on the web management interface, click the Inventory option of the **Admin** tab on the side panel.

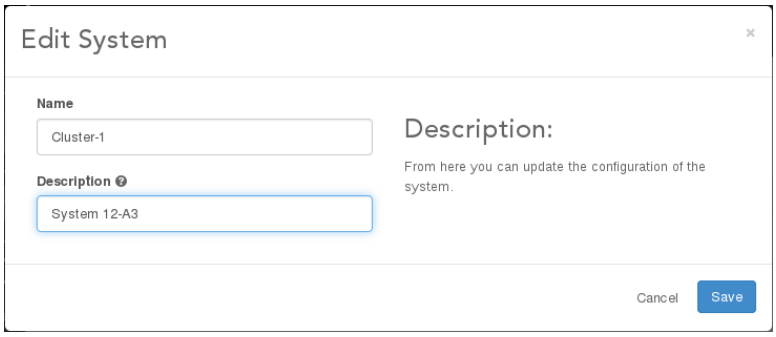
The Cluster System

The HP Helion OpenStack Carrier Grade cluster is represented internally by a unique object referred to as the **system**.

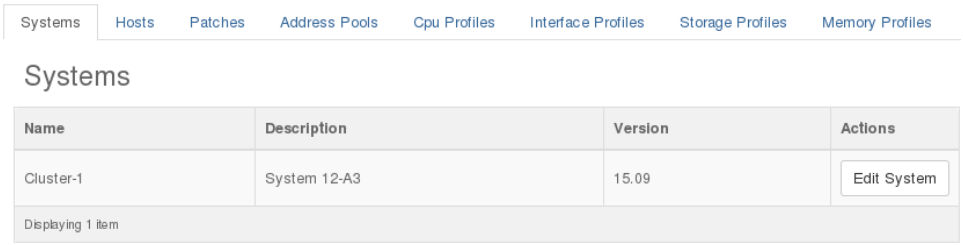
Information about this object is available from the **Systems** tab on the **System Inventory** page, as illustrated below. The software version is shown in the **Version** field.



By default, the **Name** of the system object is its internal UUID, and the **Description** is empty. To update these fields, click **Edit System**.



The **Systems** tab displays the updated information.

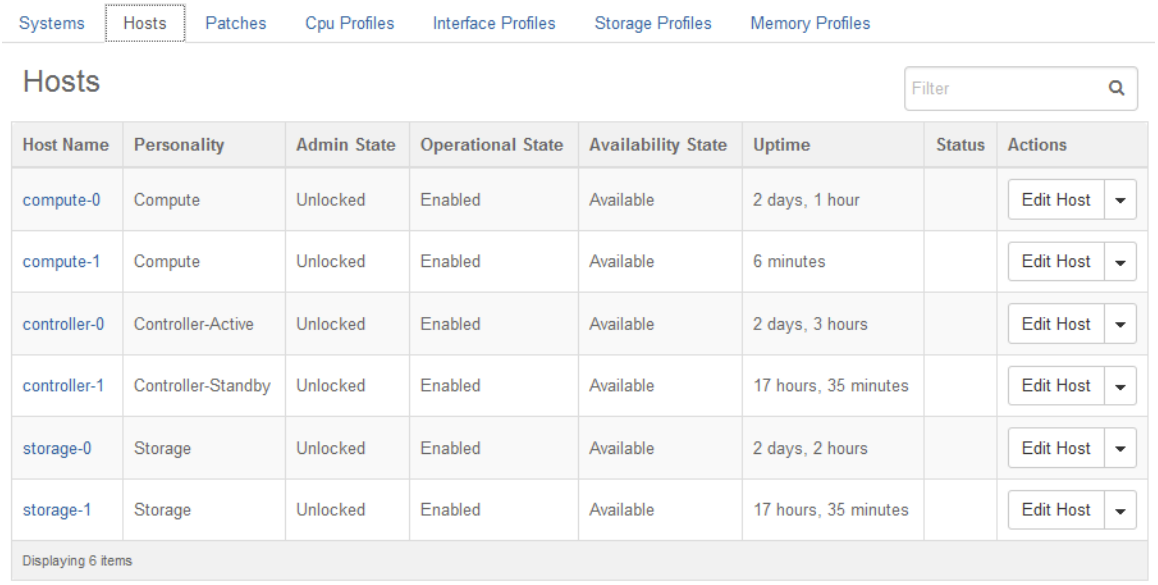


The **Name** is also displayed in the top left corner of the web management interface, and in the welcome message for an SSH login.

Host Inventory

The **Hosts** tab on the **System Inventory** page provides an overview of the current state of all hosts in the HP Helion OpenStack Carrier Grade cluster. From this tab, you can obtain detailed information about the hosts, and execute maintenance operations.

A sample hosts tab is illustrated below:



The information is refreshed periodically to reflect the ongoing changes on the cluster. It consists of the following columns:

Host Name

The name assigned to the host. This is an active link pointing to the detailed inventory page for the host. See *Inventory Detail* on page 116 for details.

Personality

The personality of the host (controller, compute, or storage).

Admin State

The administrative state of the host. It can be in one of two states:

Locked

The host is administratively prohibited from performing services. This is the initial state for hosts auto-discovered in the cluster.

A controller node in this state is not functioning in HA mode, and it is not running any active controller services.

Compute nodes in this state do not provide any service. In particular, a locked compute node is not running any virtual machine instances, and no new ones will be scheduled to run on it.

Unlocked

The host is administratively in service.

A controller node in this state, and not in the failed state, is active in its HA role, and is running the assigned controller services.

A compute node in this state, and not in the failed state, is eligible for regular scheduling and maintenance operations on virtual machines.

A storage node in this state, and not in the failed state, provides storage services.

Operational State

The operational state of the host. It can be in one of two states:

Disabled

Indicates that the host is not providing the expected services. This may be due to the fact that it is in the process of being unlocked, a failure has occurred, or it is being automatically recovered due to a failure.

Enabled

Indicates that the host is providing the expected services, even if its operational environment is compromised. In the latter case, the host is reported to be in the *degraded* availability state, in which case, state maintenance is constantly trying to recover the host to fully *available* state through in-service testing.

Availability State

The availability state of the host. It can be in one of the following states:

Offline

The host is known to HP Helion OpenStack Carrier Grade, but is not reachable for maintenance purposes.

Online

The host is reachable and ready to be unlocked.

InTest

A transient state that occurs when transitioning from locked, or from a *failed* operational state, to unlocked states. While in this state, the host is executing a series of tests to validate its hardware and software integrity.

Available

The host is fully operational and providing services.

Degraded

The host is experiencing compromised operational conditions, such as low memory, but is still providing the expected services. Details about the compromised conditions are available through the alarms subsystem. See *System Alarms* on page 171 for details.

Failed

A major fault has occurred and the host is no longer providing any services. The HP Helion OpenStack Carrier Grade maintenance system automatically tries to recover hosts in this state.

In the case of a compute node, any virtual machines that were running before are immediately evacuated to another enabled compute node with sufficient available resources.

Power-off

The host is known to have been powered off by a previous maintenance action.

Uptime

The uptime of the host, as reported by the system maintenance service.

Status

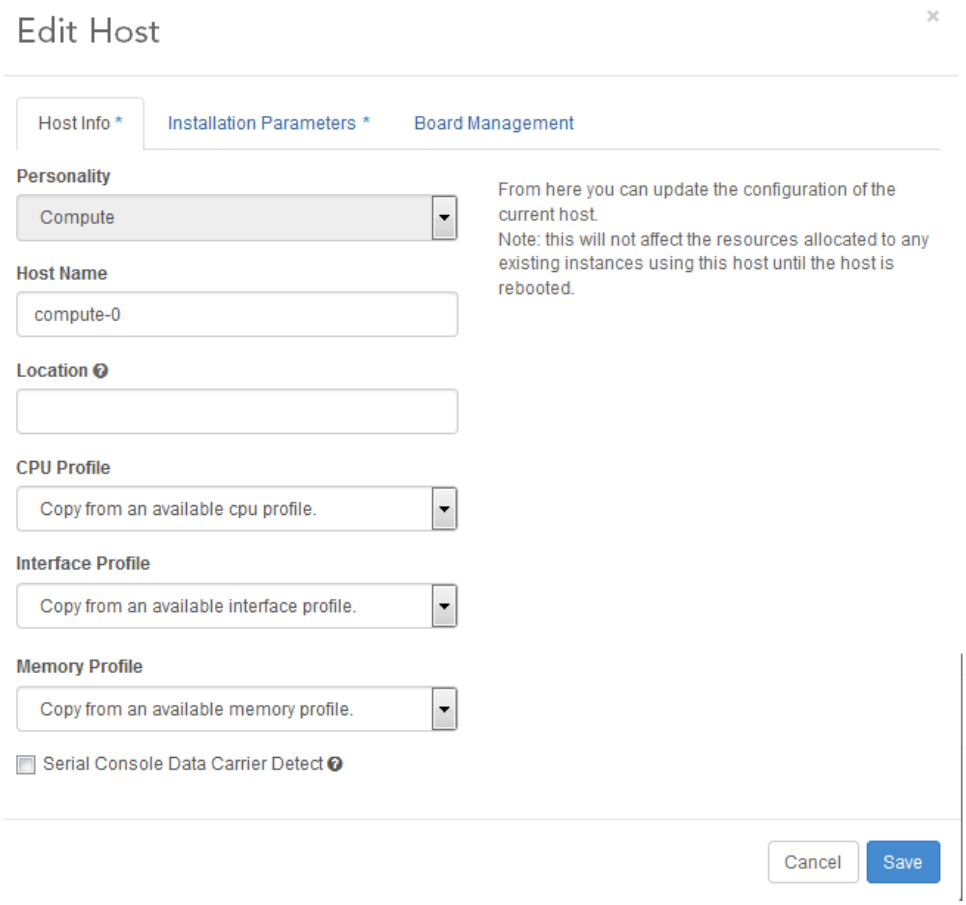
An indicator of the immediate activity occurring on the host. It reports transitory steps such as booting, initializing, configuration out of date, and in-test, which a host goes through as it transitions from one administrative or availability state to another.

Actions

The actions columns presents two buttons, as follows:

Edit Host

Clicking this button displays the **Edit Host** window as illustrated below for a compute node:



From this window you can modify the host name (compute nodes only), and apply profiles as needed. For more about profiles, see *Hardware Profiles* on page 110.

The Installation Parameters tab gives you access to installation settings. Changes take effect if the host is re-installed. For more information, see the *HP Helion OpenStack Carrier Grade Software Installation Guide*.

The **Board Management** tab gives you access to the configuration of the management board, if available on the host. See *The Board Management Network* on page 39 for details.

Note that this is the same window you use to assign the host's personality when installing the HP Helion OpenStack Carrier Grade on the host.

More (when host is locked)

This button gives you access to the list of maintenance operations that can be initiated on the host. The list is context sensitive whereby only permissible command operations are displayed.

Unlock Host

Used to bring a host into service. The first step is to reset the target host to ensure that it starts from a well-known state. The host is automatically configured, and any required software patches are applied.

The state transition only succeeds if all the necessary configuration components for the host are already in place. For example, the state transition is rejected on a compute node for which no data interfaces are defined.

You can unlock a host from the controller's command line, as follows:

```
~(keystone_admin)$  system host-unlock hostname
```

Power Off Host

Gracefully powers off the host, ensuring that all system processes are properly shut off first. This selection is available if board management is configured on the system, the host is equipped with an iLO module, and the host is in a powered-on state.

You can power off a host from the controller's command line, as follows:

```
~(keystone_admin)$  system host-power-off hostname
```

Power On Host

Powers on the host. This selection is available if board management is configured on the system, the host is equipped with an iLO module, and the host is in a powered-off state.

You can power on a host from the controller's command line, as follows:

```
~(keystone_admin)$  system host-power-on hostname
```

Reboot Host

Gracefully shuts down the host, ensuring that all system processes are properly shut off first. The host then reboots automatically.

You can reboot a host from the controller's command line, as follows:

```
~(keystone_admin)$  system host-reboot hostname
```

Reset Host

Use this selection only if **Reboot Host** fails. It performs an out-of-band reset, stopping and restarting the host without ensuring that all system processes are shut off first.

This selection is available if board management is configured on the system, the host is equipped with an iLO module, and the host is in a powered-on state.

You can reset a host from the controller's command line, as follows:

```
~(keystone_admin)$  system host-reset hostname
```

Reinstall Host

Forces a full re-installation of the HP Helion OpenStack Carrier Grade software on the host. The host's hard drive is erased, and the installation process is started afresh.

Note:

Performing a host reinstall successfully is dependent on your BIOS boot order. Observe the following tips.

- Prior to installation, the BIOS should be configured to allow booting from disk and the network.
- During the host re-installation, it is advisable to have a console serial cable attached to the host to observe the boot progress.
- BIOS boot order should be:
  1. The boot partition.

     Typically, this is the disk associated with `/dev/sda` and is as defined in `/proc/cmdline` when the load is booted.
  2. The NIC on the boot interface (e.g. management or pxe network).
- Set BIOS boot options to ensure failsafe boot if available, e.g. rotating through available boot interfaces, watchdog timer on boot, retry boot interfaces etc.

If the BIOS boot still fails to progress, it may be necessary to force boot via the network interfaces via the BIOS boot option.

Delete Host

Removes the host from the inventory database, and its hard drive is erased.

You can delete a host from the controller's command line, as follows:

```
~(keystone_admin)$  system host-delete hostname
```

Install Patches

Initiates any pending patching operations. Once successfully executed, the host returns back to the locked state. See *Managing Software Patches* on page 193 for details.

This option is only available if there are patching operations pending for the host.

More (when host is unlocked)

This button gives you access to the list of maintenance operations that can be initiated on the host. The list is context sensitive whereby only permissible command operations are displayed.

Lock Host

Attempts to bring the host out of service.

On a controller node, the state transition only succeeds if there are no services running in active mode on the host.

On a compute node, the state transition only succeeds if all currently running instances on the host can be migrated to alternative compute nodes. Migration of the virtual machine instances is initiated automatically as soon as the state transition is requested.

Note:

Live migration is not currently supported for an instance using local storage. Locking the host initiates a cold migration. The local storage for the instance is rebuilt.

You can lock a host from the controller's command line, as follows:

```
~(keystone_admin)$  system host-lock hostname
```

Forced Lock Host

Forces the host to be out of service. Use this option only if for some reason you cannot bring the host out of service using the Lock Host option. When selected, the system displays a warning message appropriate to the personality of the host. Use this option with caution.

You can force lock a host from the controller's command line, as follows:

```
~(keystone_admin)$  system host-lock --force hostname
```

Note that a force lock operation on a compute node causes an immediate service outage on all hosted virtual machines.

Swact Host

This operation is available on controller nodes only, and should be run on the active controller to initiate a switch of the active/standby roles.

Swact is an abbreviated form of the term **Switch Active** (host). When selected, this option forces the other controller to become the active one in the HA cluster. This means that all active system services on this controller move to standby operation, and that the corresponding services on the other controller become active.

Use this option when you need to lock the currently active controller, or do any kind of maintenance procedures, for example, when updating hardware or replacing faulty components.

You can swact a host from the controller's command line, as follows:

```
~(keystone_admin)$  system host-swact --force hostname
```

Install Patches

Initiates any pending patching operations. Once successfully executed, the host returns back to the unlocked state. See *Managing Software Patches* on page 193 for details.

This option is only available if there are patching operations pending for the host.

The Life Cycle of a host

The life cycle of a host is the set of state transitions the host goes through as its current state changes. The host states in HP Helion OpenStack Carrier Grade are based on the *ITU X.731 State Management Function Specification for Open Systems*.

The current state of a host is determined by the allowed combinations of the administrative, operational, and availability states at any given time. The following figure illustrates the life cycle of a host.

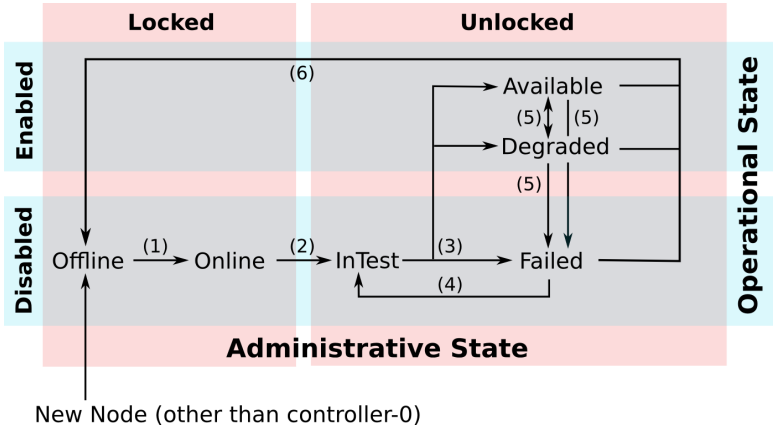


Figure 4: The Life Cycle of a Host

In this figure:

- the administrative states, locked and unlocked, are presented in two columns
- the operational states, disabled and enabled, are presented in two rows
- the availability states are presented as elements inside the administrative/operational matrix

The description that follows uses the availability states only, since for each state the corresponding administrative and operational states can be read directly from the figure.

The life cycle of a new host starts when it is discovered by the active controller on the internal management network. A new host is initially reported as *Offline*. As an exception, the first controller, **controller-0**, is automatically set to *available* during initial commissioning. The following are the available transitions. Numbers are attached to them for easier reference:

(1) *Offline* to *Online*

This transition takes place once the administrator configures the host name and personality of the host. During the transition, the HP Helion OpenStack Carrier Grade software is installed and the host reboots. The transition concludes when the controller establishes maintenance and inventory connectivity with the new host.

(2) *Online* to *InTest*

This transition takes place when the administrator requests to move the host from the locked to the unlocked administrative states. The host reboots first. After it finishes booting, it establishes maintenance communication and enters the transient **InTest** state. While in this state, the configuration is applied, and a set of hardware and software tests are executed to ensure the integrity of the host.

(3) *InTest* to *Available*, *Degraded*, or *Failed*

The transition is initiated automatically after the activities in the transient state **inTest** are complete. Depending on the outcome, the host goes into one the three states.

(4) *Failed* to *InTest*

This is a value-added maintenance transition that the HA framework executes automatically to recover failed hosts.

(5) *Available* to/from *Degraded*, *Available* to *Failed*, and *Degraded* to *Failed*

These are transitions that may occur at any time due to changes on the operational state and faults on unlocked hosts.

(6) *Available*, *Degraded*, or *Failed*, to *Offline*

These are maintenance transitions that take place automatically to reflect the operational state of a host.

On a compute node in *Available* or *Degraded* state, the transition triggers the migration of the active instances to another available compute node.

Hardware Profiles

A hardware profile is a named object that captures a particular host configuration. The profile can be applied to any other host with a similar hardware configuration.

The following types of hardware profiles are available from the **Systems Inventory** page, each presented on a separate tab:

- CPU profiles
- interface profiles
- storage profiles
- memory profiles

Once a hardware profile is defined, it can be applied to any host where it makes sense to be applied to change its configuration.

CPU Profiles

A CPU profile is a named assignment of processors and cores to one or more of the following types of processing threads:

Platform

System threads handling the core hosting functionality of the host. Platform threads are always present on all types of hosts in the cluster.

vSwitch

AVS threads dedicated to handling network traffic and packet processing tasks. They exist on the compute nodes only.

Shared

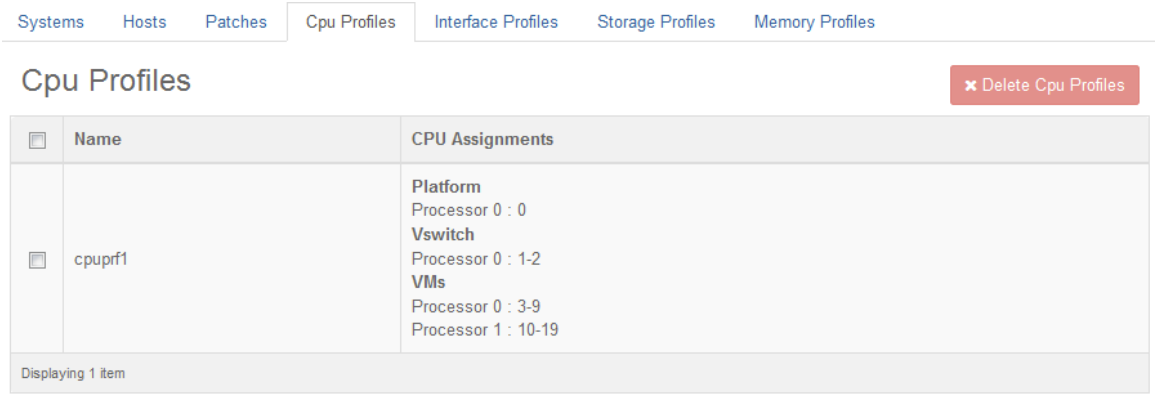
Threads handling low-load or non-real-time virtual machine tasks, implemented on a shared physical CPU in a compute node.

Virtual Machines

Threads handling virtual machines. They exist on the compute nodes only.

On controller nodes, all processors and cores are automatically assigned to platform threads, as they are the only ones available. On compute nodes, which always run a number of threads of all types, a default CPU allocation is done automatically when the system software is initially installed.

A list of the currently defined CPU profiles is available from the **CPU Profiles** tab on the **System Inventory** page, as illustrated below.



This example lists one CPU profile, named **HP-360-C1**, with the following processor and core allocation:

Table 3: CPU Allocation Example

Thread Type	CPU Allocation
Platform	Processor 0, core 0
vSwitch	Processor 0, cores 1 and 2
Virtual Machines	Processor 0, cores 3 to 7 Processor 1, cores 8 to 15

CPU profiles can be deleted using the button **Delete CPU Profiles** on this page. When clicked, it deletes all check-marked profiles in the list. The delete operation does not affect CPU profiles already applied.

See *Processor* on page 118 for more information of how to change the CPU allocation for a compute node, and on how to define CPU profiles.

Interface Profiles

An interface profile is a named configuration of Ethernet ports and interfaces on a host, as follows:

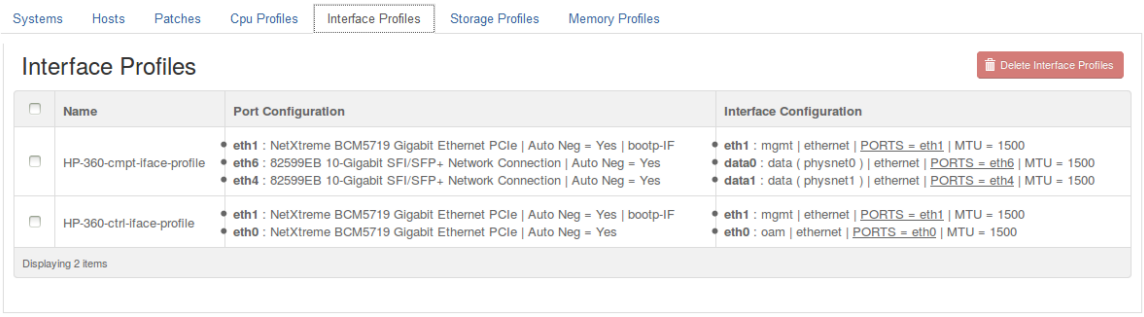
Ethernet Ports

The list of physical Ethernet ports that have been allocated to connect to a network.

Interfaces

Logical L2 interfaces defined on top of physical Ethernet ports.

A list of the currently defined interface profiles is available from the **Interface Profiles** tab on the **System Inventory** page, as illustrated below.



This example lists two interface profiles, as follows:

Table 4: Interface Profiles

Profile Name	Ports	Interfaces
HP-360-cmpt-iface-profile	eth1, eth4, eth6	eth1 on internal management network data0 (eth6 on physnet0) data1 (eth4 on physnet1)
HP-360-ctrl-iface-profile	eth0, eth1	eth0 on OAM network eth1 on internal management network

The interface profile **HP-360-cmpt-iface-profile** belongs to a compute node. It lists three physical ports, eth1, eth4, and eth6, allocated to connect to a network. It also lists three logical L2 interfaces:

- eth1, allocated to connect to the internal management network
- data0, a data interface associating physical port eth6 with the provider network physnet0.
- data1, a data interface associating physical port eth4 with the provider network physnet1.

The interface profile **HP-360-ctrl-iface-profile** belongs to a controller node. It lists two physical ports, eth0, and eth1, and two logical L2 interfaces connecting these ports to the OAM and internal management networks.

Interface profiles can be deleted using the button **Delete Interface Profiles** on this page. When clicked, it deletes all check-marked profiles in the list. The delete operation does not affect interface profiles already applied.

See *Interfaces* on page 131 for more information on how to create interface profiles.

Storage Profiles

A storage profile is a named configuration for a list of storage resources on a computing node. Each storage resource consists of the following elements:

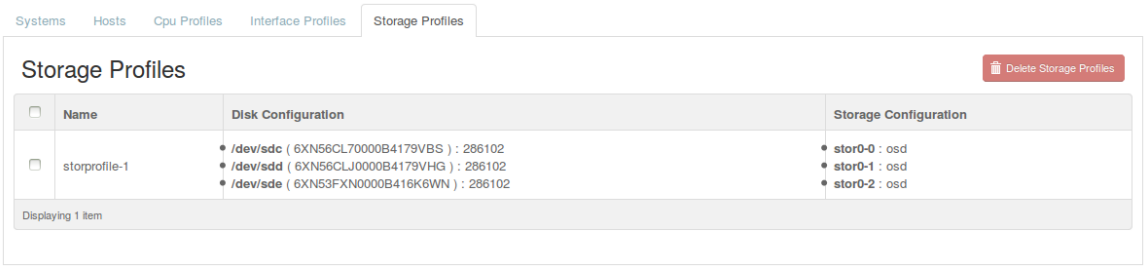
Disks

A Linux block storage device, such as `/dev/sdd`, identifying an entire hard drive.

Storage Volumes

A storage volume consisting of a name and a storage function. The name is used as a human-readable version of the native storage device UUID. The storage function indicates the type of storage backend, such as **OSD** for a storage system.

A list of the currently defined storage profiles is available from the **Storage Profiles** tab on the **System Inventory** page, as illustrated below.



This example lists one storage profile named **storprofile-1** with three storage resources on the hard drives `/dev/sdc`, `/dev/sdd`, and `/dev/sde`. All storage resources are of the **OSD** type.

See *Storage* on page 126 for more information on how to create storage profiles.

Memory Profiles

A memory profile is a named assignment of huge pages for use as VM memory. For more information, see *Creating and Using Memory Profiles* on page 125.

Host Management on an Active System

You can add or remove hosts and change disks without interrupting HP Helion OpenStack Carrier Grade.

To make changes to a host, you must lock it. Locking a host automatically and seamlessly redirects activity to other available hosts. While the host is locked, you can safely power it down and make disk changes, or remove the host and insert a new one.

After making the required configuration changes, you can unlock the host and begin using it.

Managing Controller Nodes

You can replace controller nodes or disks while the system is running.

The HP Helion OpenStack Carrier Grade system uses exactly two controllers; you cannot add or remove a controller. However, you can replace the primary or secondary disks, and you can replace faulty controller nodes.

Note:

If you are replacing disks in order to increase the controller storage capacity, follow the instructions for *Changing Storage Space Allotments on the Controller* on page 85.

1. Lock the standby controller.

   Open the **Hosts** list by clicking **Inventory** on the **System Panel** section of the **Admin** menu, and then selecting the **Hosts** tab.

   Click **More** > **Lock Host** for the controller.

   Wait for the procedure to be completed.
2. Power down the standby controller and make any required hardware changes.

   This may involve replacing disks, or replacing the host completely.
3. Place the new or modified controller into service.

   a) Power up the controller.

   Wait until the controller is reported as **Locked**, **Enabled**, and **Available**.

   b) If required, reinstall the HP Helion OpenStack Carrier Grade software on the controller.

   If you are making disk changes to increase storage capacity, you must re-install the HP Helion OpenStack Carrier Grade software. For more information, see *Changing Storage Space Allotments on the Controller* on page 85.

To reinstall the software on a host , click **More** > **Reinstall**.

Note:

Performing a host reinstall successfully is dependent on your BIOS boot order. Observe the following tips.

- Prior to installation, the BIOS should be configured to allow booting from disk and the network.
- During the host re-installation, it is advisable to have a console serial cable attached to the host to observe the boot progress.
- BIOS boot order should be:
  1. The boot partition.

     Typically, this is the disk associated with `/dev/sda` and is as defined in `/proc/cmdline` when the load is booted.
  2. The NIC on the boot interface (e.g. management or pxe network).
- Set BIOS boot options to ensure failsafe boot if available, e.g. rotating through available boot interfaces, watchdog timer on boot, retry boot interfaces etc.

If the BIOS boot still fails to progress, it may be necessary to force boot via the network interfaces via the BIOS boot option.

Wait for the host to be reported as **Locked**, **Disabled**, and **Online**.

c) Perform a swact.

Click **More** > **Swact Host** for the active controller.

The standby controller becomes the active controller, and the original active controller is placed into standby.

4. Lock the original active controller (now in standby).

   Open the **Hosts** list by clicking **Inventory** on the **System Panel** section of the **Admin** menu, and then selecting the **Hosts** tab.

   Click **More** > **Lock Host** for the controller.

   Wait for the procedure to be completed.
5. Power down the controller and make the same hardware changes.

   Caution:

   The configurations for controller-0 and controller-1, including disk types and sizes, must be identical.
6. Power up the new or modified controller.
7. If required, reinstall the HP Helion OpenStack Carrier Grade software on the controller.

The updated controllers are now in service. The controller that was formerly active is now the standby controller.

Compute Node Management

The compute nodes in HP Helion OpenStack Carrier Grade form a resource pool for hosting guest instances. You can manage this pool by managing the hosts.

You can change the resource pool in several ways:

- You can add or remove hosts to increase or decrease the size of the pool.
- You can replace a host with another that has different resources (for example, memory, or number of CPU cores).
- You can adjust the resources on an existing host.
- You can replace a failed compute node host with an equivalent.

Caution:

When replacing or adjusting a host, ensure that the overall resource pool still meets the requirements for your system.

Complete instructions for adding a compute node are provided in the *HP Helion OpenStack Carrier Grade Software Installation Guide: Initializing and Configuring Compute Nodes*.

Removing Compute Nodes

You can remove a compute node from the pool of available resources.

You may need to remove a compute node in order to replace a failed host, or to change the configuration of a host. If the host is active, you can migrate instances on it by locking the host.

Caution:

Before locking a host, ensure that sufficient resources are available on other hosts to migrate any running instances.

1. Lock the host to be removed.

   Open the **Hosts** list by clicking **Inventory** on the **System Panel** section of the **Admin** menu, and then selecting the **Hosts** tab.

   Click **More** > **Lock Host** for the host.

   Wait for the procedure to be completed.
2. Delete the host from the HP Helion OpenStack Carrier Grade system.

   Click **More** > **Delete Host** for the host.
3. Power down the host and remove the hardware from the cluster.

Adjusting Resources on a Compute Node

You can adjust the resources of a compute node while it is offline.

1. Lock the host to make changes.
   a) On the **Admin** menu of the web administration interface, open the **System Panel** section, and then select **Inventory**.
   b) Select the **Hosts** tab.
   c) Open the **More** drop-down list for the host, and then select **Lock Host**.
   d) Wait for the host to be reported as **Locked**.
2. Power off the host.
3. Make any required resource changes (for example, BIOS changes required for proper operation).

   If you are adding a disk to provide additional local storage for VMs, you can install an unpartitioned disk. New disks are detected by the compute node operating system and automatically configured with a single partition.
4. Power on the host, and wait for it to reboot fully.

   When the host is fully rebooted, it is shown as **Locked**, **Enabled**, and **Available** in the **Hosts** list.

   Note:

   Do not unlock the host until it is fully rebooted.
5. Unlock the host.

   The host is rebooted a second time.

When the host is reported as **Unlocked**, **Enabled**, and **Available**, it is ready for use with the adjusted resources.

Inventory Detail

From the **Inventory Detail** page you can see detailed information about a host, and use it to define hardware profiles that can be used across the cluster.

You access the **Inventory Detail** page by clicking the host name on the **Hosts** tab of the **Systems Inventory** page, as described in *Managing Hardware Inventory* on page 103. The inventory detail for a host consists of multiple tabs, each addressing a different aspect of the host. They include:

- overview
- processor
- memory
- storage
- ports
- interfaces
- sensors

Overview

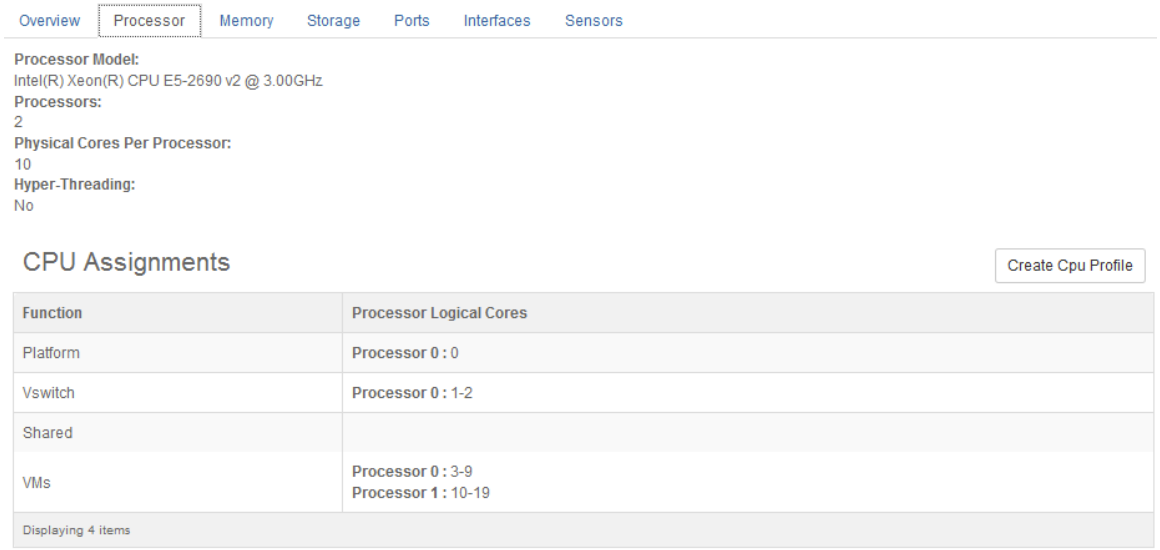
The **Overview** tab on the **Inventory Detail** page summarizes core details about a host object.

The following items are included in the summary:

- host name, personality, and the internal UUID and host ID reference numbers
- MAC and IP addresses on the internal management network
- serial ID, if known. Use the command `system host-modify <hostname> serialid=xxx` to update it.
- location, as entered by the operator using the **Edit Host** window (see *Host Inventory* on page 103)
- time stamps of when the host was created and last updated
- the host's state
- board management (iLO) information, if available, including controller type, MAC address, and IP address.

Processor

The **Processor** tab on the **Inventory Detail** page presents processor details for a host, as illustrated below.



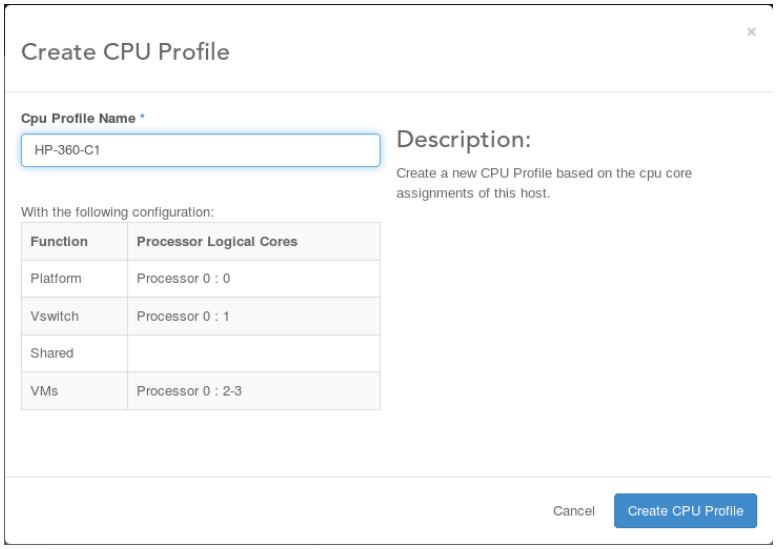
The **Processor** tab includes the following items:

- processor model, number of processors, number of cores per processor, and Hyper-Threading status (enabled or disabled).
- the CPU assignments. See section *CPU Profiles* on page 110 for more details.

Two buttons are also available as follows:

Create CPU Profile

Clicking this button displays the **Create CPU Profile** window, where the current CPU assignment can be given a name, as illustrated below.

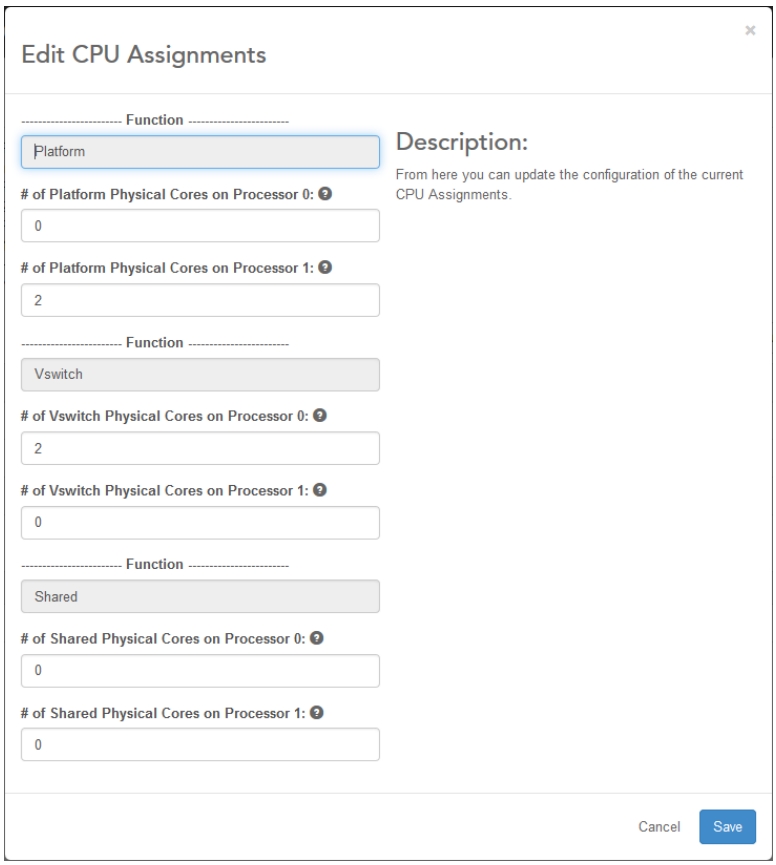


In this example, you are about to create a CPU Profile named **HP-360-C1** out of the current CPU assignments found in a compute node.

Edit CPU Assignments

This button is available only when the host is in the locked state.

Clicking this button displays the **Edit CPU Assignments** window. On a compute node, you can use this window to assign cores to specific functions, as illustrated below. Unassigned cores are available for allocation to virtual machine threads.



Note:

On a controller or storage node, only the **Platform** function is shown, and all available cores are automatically assigned as platform cores.

Platform

You can reserve one or more cores in each NUMA node for platform use. One core on each host is required to run the operating system and associated services.For a system with separate controller and compute nodes, one core on each host is required to run the operating system and associated services. For a combined controller and compute node in a minimal two-server configuration, two cores are required.

The ability to assign platform cores to specific NUMA nodes offers increased flexibility for high-performance configurations. For example, you can dedicate certain NUMA nodes for vSwitch or VM use, or affine VMs that require high-performance IRQ servicing with NUMA nodes that service the requests.

Vswitch

AVS (vSwitch) cores can be configured for each processor independently. This means that the single logical vSwitch running on a compute node can make use of cores in multiple processors, or NUMA nodes. Optimal data path performance is achieved when all AVS cores, the physical ports, and the virtual machines that use them are running on the same processor. You can affine VMs to NUMA nodes with AVS cores; for more information, see *Affining a VM to a NUMA Node with a vSwitch Core* on page 151. Alternatively, having AVS cores on all processors ensures that all virtual machines, regardless of the core they run on, are efficiently serviced. The example allocates two cores from processor 1 to the AVS threads.

Shared

One physical core per processor can be configured as a shared CPU, which can be used by multiple VMs for low-load tasks. To use the shared physical CPU, each VM must be configured with a shared vCPU ID. For more information, see *Pinning a vCPU to a Shared Physical CPU* on page 146.

Changes do not take effect until the host is locked.

To see how many cores a processor contains, hover over the Information icon.

Viewing NUMA Node Resources on a Host

You can use the CLI to display the NUMA node resources for a host.

Host NUMA nodes can be *pinned*, or assigned for use by VMs. For example, a VM can be configured to use NUMA node 0, so that when the VM is launched or migrated, the virtual machine scheduler locates a host node with an available NUMA node 0, and dedicates that NUMA node for use by the VM. For more about pinning NUMA nodes, see *Pinning a Guest NUMA Node to a Host NUMA Node* on page 151.

The resources of the pinned NUMA node, including the number of available CPUS and the available memory, must be sufficient to meet the requirements of the VM, which can be specified independently. (For more about specifying NUMA node requirements for a VM, see *Configuring the NUMA Node Allocations for a VM* on page 149.) To ensure that a given host NUMA node can support the VM requirements, you can review the CPU and memory complements for host NUMA nodes.

To view the CPU complement for a NUMA Node (that is, for a socketed physical processor), use the vm-topology command. For details, see *Processor* on page 118.

Designating Shared Physical CPUs on a Compute Host

You can designate one shared physical CPU per physical processor on a compute host to run low-load or non-real-time tasks for multiple VMs, freeing other cores on the host for dedicated high-load tasks.

You can use the web administration interface or the CLI to set up shared physical CPUs.

To add or remove a shared physical CPU from the CLI, use a command of the following form:

```
~(keystone_admin)$ system host-cpu-modify -f shared -
pprocessor use_shared hostname
```

where

processor

is the number of the physical processor (0 or 1)

use_shared

specifies whether to use a shared physical CPU (0 for no, 1 for yes)

hostname

is the name of the compute host

For example, to set up a shared physical CPU on processor **0** of **compute-0**:

```
~(keystone_admin)$ system host-cpu-modify -f shared -p0 1 compute-0
```

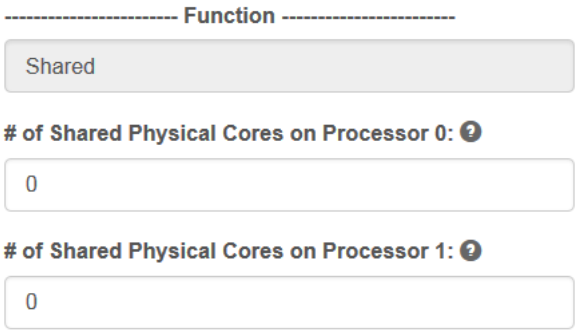
1. Lock the host to make changes.
   a) On the **Admin** menu of the web administration interface, open the **System Panel** section, and then select **Inventory**.
   b) Select the **Hosts** tab.

c) Open the **More** drop-down list for the host, and then select **Lock Host**.
d) Wait for the host to be reported as **Locked**.

2. Open the **Inventory Detail** page for the locked host.

   In the **Host Name** column, click the name of the host.
3. Select the **Processor** tab.
4. Click **Edit CPU Assignments**.

   This control is available only if the host is locked.
5. In the **Edit CPU Assignments** dialog box, use the **Shared** Function section to enable shared physical CPUs.



   You can designate one core on each physical processor for use as a shared physical CPU. The actual core is assigned from the pool of available cores for the processor.

   For example, to use a core on processor 0 as a shared physical CPU, set the **# of Shared Physical Cores on Processor 0** to 1. Valid values are **1** (to assign a core as a shared physical CPU) or **0** (if a shared physical CPU is not required on the processor.)
6. Lock the host to make the changes take effect.

To configure a VM to use a shared physical CPU, see *Pinning a vCPU to a Shared Physical CPU* on page 146.

Changing the Hyper-threading Status

The hyper-threading status is controlled by the BIOS settings of the host.

1. Lock the host to prepare it for configuration changes.

   In the **Hosts** list, click **More** for the host, and then select **Lock Host**.

   The host is locked and reported as **Locked**, **Disabled**, and **Online**.
2. Edit the host BIOS settings to enable or disable hyper-threading.

   For more about editing the BIOS, refer to the documentation provided by the maker of the host computer.
3. Unlock the host to make it available for use.

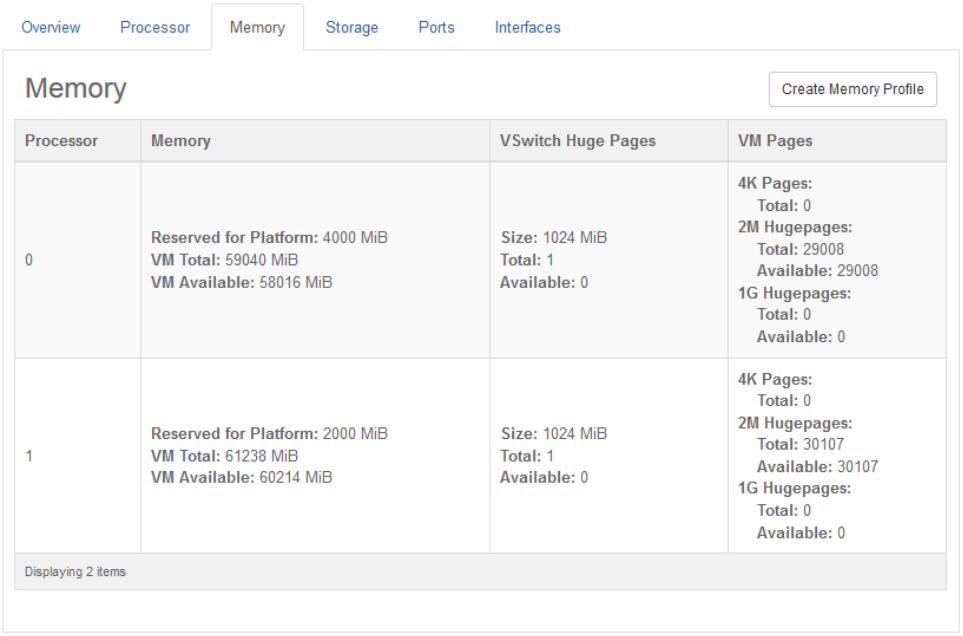
   In the **Hosts** list, on the row associated with the node, click **More** > **Unlock Host**.

   The host is rebooted, and its **Availability State** is reported as **In-Test**. After a few minutes, it is reported as **Unlocked**, **Enabled**, and **Available**.
4. Confirm the hyper-threading status in HP Helion OpenStack Carrier Grade.

   The hyper-threading status is reported on the **Processor** tab for the host. For more information, see *Processor* on page 118.

Memory

Visit the **Memory** tab on the **Inventory Detail** page to display host memory details.



The information is presented in three columns, as follows:

Memory

Overall memory on the host.

For a controller node it displays the total and available memory figures.

For a compute node, as in the example above, it displays the amount of memory reserved for the platform (system software), and the total and available figures for use by virtual machines.

VSwitch Huge Pages

This column is relevant on compute nodes only.

The size of the huge pages, and the total and available huge page figures.

VM Pages

This column is relevant on compute nodes only.

The size of the pages, and the total and available page figures. If changes to the huge page allocations have been requested for a locked host, they are shown as **Pending**.

Host Memory Provisioning

For each NUMA node on a host, you can adjust the amount of memory reserved for platform use, and the size and number of memory pages allocated for use by VMs.

Memory not reserved for platform use is made available as VM memory. By default, the VM memory is partitioned using 2 MiB huge pages. You can change this for individual NUMA nodes to use a combination of 2 MiB, and 1 GiB huge pages. Using larger pages can reduce page management overhead and improve system performance for systems with large amounts of virtual memory and many running instances.

You can use the `system host-memory-list` and `system host-memory-show` commands to see how much memory is available for VMs. This information is also shown on the **Memory** tab of the **Host Inventory** page (see *Memory* on page 122).

After setting the memory allocations for a host, you can save them as a *memory profile*, and then apply the profile to other hosts. For more information, see *Creating and Using Memory Profiles* on page 125.

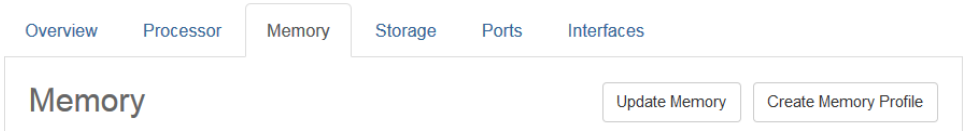
For individual VMs, you can specify which page size to use. For more information, see *Specifying a Page Size for a VM* on page 152.

Allocating Host Memory for VM Pages or Platform Use

You can edit the platform and VM page memory allocations for a NUMA node from the web administration interface using the **Memory** tab on the **Inventory** pane.

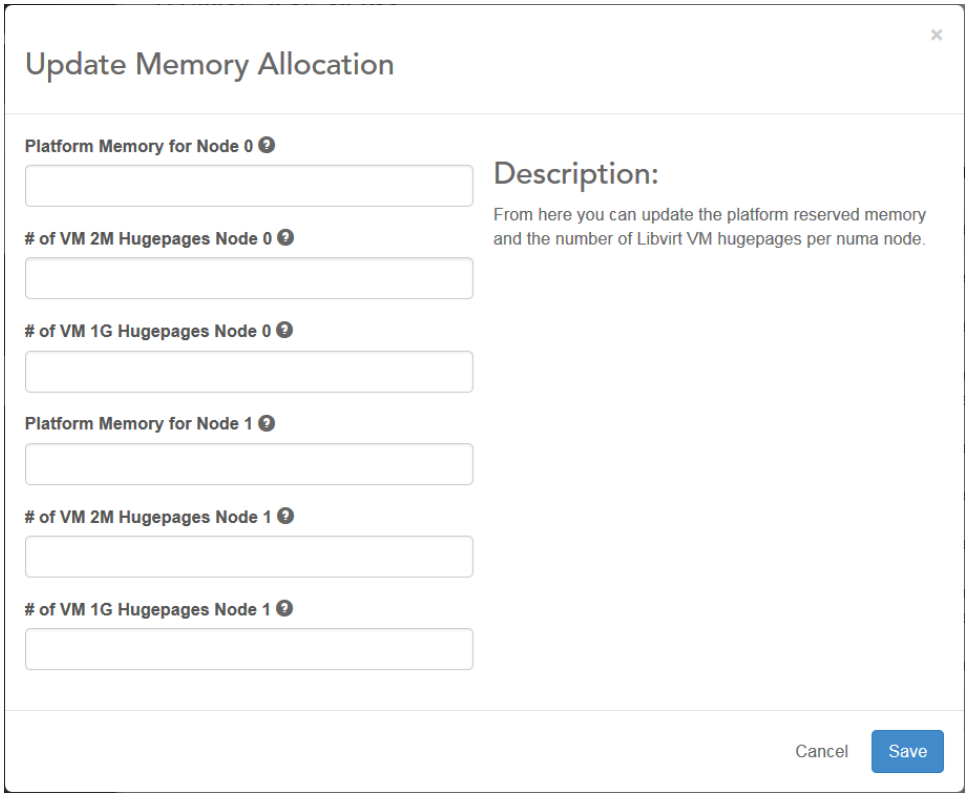
Before requesting huge pages on a host, ensure that the host has enough available memory. For details, see *Host Memory Provisioning* on page 122. If a huge page request cannot be allocated from the available memory, an informative message is displayed.

1. On the **Inventory** pane, lock the host you want to edit.
2. Click the **Host Name** to open the settings for the host.
3. On the **Memory** tab, click **Update Memory**.



4. Use the **Update Memory Allocation** dialog box to set the memory allocations for each NUMA node.

   For each available NUMA node, three fields are supplied, as illustrated in the following example screen for two NUMA nodes.



Platform Memory for Node *n*

The amount of memory to reserve for platform use on the NUMA Node, in MiB. To see the minimum requirement, hover over the Information icon next to the field.

# of VM 2M Hugepages Node *n*

The number of 2 MiB huge pages to reserve for VM use on the NUMA Node. If no 2 MiB pages are required, type 0.

# of VM 1G Hugepages Node *n*

The number of 1 GiB huge pages to reserve for VM use on the NUMA Node. If no 1 GiB pages are required, type 0.

To see how many huge pages of a given size you can successfully request on a node (assuming that pages of another size are not also requested), hover over the Information icon next to the field.

Any unused memory is automatically allocated as 4 KiB pages of regular memory for VMs. You must use the Memory Page Size extra spec, or suitable image metadata, to use these 4 KiB pages as the backing mechanism for VM memory allocation requests. See *Specifying a Page Size for a VM* on page 152 for details.

5. Click **Save**.
6. Unlock the host and wait for it to be reported as **Available**.

Allocating Host Memory Using the CLI

You can edit the platform and huge page memory allocations for a NUMA node from the CLI.

1. Lock the affected host.

```
(keystone_admin)$ system host-lock hostname
```

2. Use the following command to set the memory allocations.

```
(keystone_admin)$ system host-memory-modify hostname processor [-
m reserved] [-2M 2Mpages] [-1G 1Gpages]
```

where

hostname

is the hostname or id of the compute node

processor

is the NUMA node of the compute node (0 or 1)

reserved

(if the optional -m argument is included) the amount of memory reserved for platform use, in MiB.

2Mpages

(if the optional -2M argument is included) the number of 2MiB huge pages to make available.

1Gpages

(if the optional -1G argument is included) number of 1GiB huge pages to make available

For example, to allocate four 2 MiB huge pages for use by VMs on NUMA node 1 of compute node **compute-0** :

```
(keystone_admin)$ system host-memory-modify compute-0 1 -2M 4
```

3. Unlock the host.

```
(keystone_admin)$ system host-unlock hostname
```

4. Wait for the host to be reported as **available**.

```
(keystone_admin)$ system host-list hostname
+----+--------------+-------------+----------------+-------------
+--------------+
| id | hostname     | personality | administrative | operational |
 availability |
+----+--------------+-------------+----------------+-------------
+--------------+
| 1  | controller-0 | controller  | unlocked       | enabled     |
 available    |
| 2  | controller-1 | controller  | unlocked       | enabled     |
 available    |
| 3  | compute-0    | compute     | unlocked       | enabledd    |
 available      |
+----+--------------+-------------+----------------+-------------
+--------------+
```

Creating and Using Memory Profiles

You can optionally save the memory allocations for a host as a profile, and apply the profile to other hosts.

For more information about host memory allocation, see *Host Memory Provisioning* on page 122.

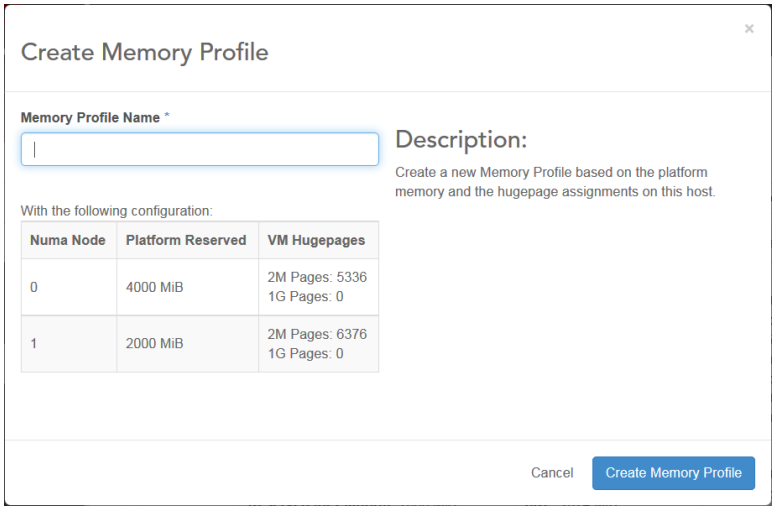
As an alternative to the web administration interface, you can use the following CLI command to create a memory profile:

```
~(keystone-admin)$ system memprofile-add memoryprofile hostid
```

where *memoryprofile* is the name or UUID of the memory profile, and *hostid* is the name or UUID of the host from which to create the profile.

1. Open the **Inventory Detail** page for the host.
   a) On the **Admin** menu of the web administration interface, open the **System Panel** section, and then select **Inventory**.
   b) Select the **Hosts** tab, and then in the **Host Name** column, click the name of the host.
2. On the Memory tab, click **Create Memory Profile**.

   The **Create Memory Profile** dialog box appears.



3. Type a name for the memory profile.
4. Click **Create Memory Profile** to save the profile and close the dialog box.

You can apply this profile to other hosts by editing the host settings in the web administration interface. For information about editing a host, see *Host Inventory* on page 103. You can also use the following CLI command to apply a memory profile to a host:

```
~(keystone-admin)$ system host-apply-memprofile hostid memoryprofile
```

where *hostid* is the name or UUID of the host, and *memoryprofile* is the name or UUID of the memory profile.

To manage memory profiles, you can use the **Memory Profiles** tab on the **System Inventory** page, or you can use the following CLI commands:

- To list memory profiles:

  ```
  ~(keystone-admin)$ system memprofile-list
  ```

- To show details for a memory profile:

  ```
  ~(keystone-admin)$ system memprofile-show memoryprofile
  ```

  where *memoryprofile* is the name or UUID of the memory profile.

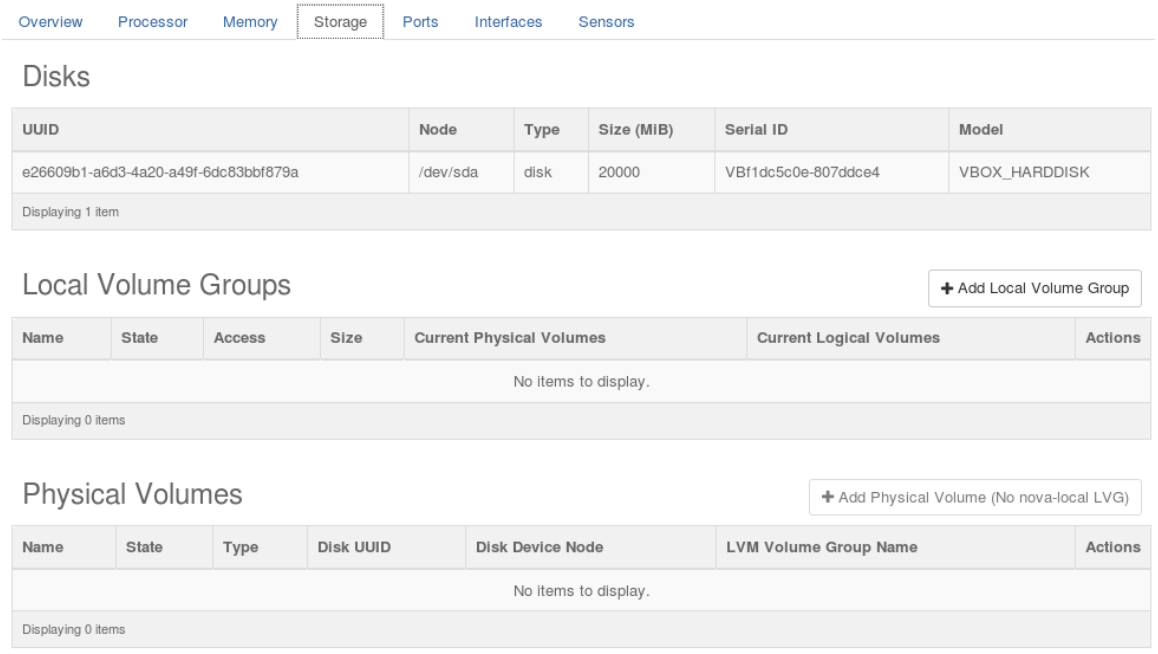
- To delete a memory profile:

  ```
  ~(keystone-admin)$ system memprofile-delete memoryprofile
  ```

  where *memoryprofile* is the name or UUID of the memory profile.

Storage

The **Storage** tab on the **Inventory Detail** page presents storage details for a host.



The information is presented in one or more lists, as determined by the host type.

Disks

This list is presented for all host types. It lists all available hardware devices used for storage. For each device, the following information is included:

UUID

The unique identifier for the device.

Node

The Linux device name.

Type

The type of storage device, typically a hard drive (disk).

Size

The capacity of the device in MiB.

Serial ID

The device's serial ID number.

Model

The manufacturer's model for the device.

Storage Volumes

This list is presented for storage hosts. It shows a list of logical storage volumes defined on available disks. For each volume, the following information is included:

UUID

The unique identifier for the storage volume.

Name

The name assigned to the volume, if any.

Function

The type of storage backend handling the storage volume.

For information about creating storage volumes or adding storage profiles, see the *HP Helion OpenStack Carrier Grade Software Installation Guide: Creating Storage Volumes*.

Local Volume Groups

This list is presented for compute nodes. It shows groups that provide local storage for use by VMs. For more information, see *Managing Local Volume Groups* on page 128.

For each group, the following information is provided:

Name

The name assigned to the local volume group.

State

The availability of the local volume group.

Access

The access status of the volume group (**w**riteable, **r**eadonly, resi**z**eable, e**x**ported, **p**artial, or **c**lustered).

Size

The capacity of the device in bytes.

Current Physical Volumes

The number of physical volumes that define the local volume group.

Current Logical Volumes

the number of logical volumes contained by the local volume group.

Actions

Available actions that can be performed on the local volume group.

Physical Volumes

This list is presented for compute nodes. It shows physical volumes that provide local storage for use by VMs. For more information, see *Managing Physical Volumes* on page 129.

For each group, the following information is provided:

Name

The device name associated with the physical volume.

State

The availability of the physical volume.

Type

The device type used for the physical volume.

Disk UUID

the unique identifier of the disk used to implement the physical volume.

Disk Device Node

The device used to implement the physical volume.

LVM Logical Group Name

The name of the local volume group to which the physical volume belongs.

Actions

Available actions that can be performed on the physical volume.

Managing Local Volume Groups

You can add, delete, and review local volume groups on a compute host.

Local volume groups are used to designate physical volumes on a compute host as local storage for use by VMs. You can use the web administration or the CLI to manage them. For web administration interface instructions, see *Configuring a Compute Host to Provide Local Storage* on page 64.

Before you can modify the settings for a host, you must lock the host:

```
~(keystone_admin)$ host-lock hostname
```

To configure a compute host to provide local storage for use by VMs, add a local volume group with the name **nova-local**.

The following CLI commands are available for managing local volume groups.

```
~(keystone_admin)$ host-lvg-list hostname
```

```
~(keystone_admin)$ host-lvg-show hostname
```

```
~(keystone_admin)$ host-lvg-add hostname groupname
```

```
~(keystone_admin)$ host-lvg-delete hostname groupname
```

```
~(keystone_admin)$ host-lvg-modify hostname groupname -l nova-local -i size
```

where

groupname

is the name of the local volume group

Note:

The only valid **groupname** is **nova-local**.

size

is the space in MB to allot for instances logical volume space..

To complete the configuration of a compute host for local storage, you must also add physical volumes to the **nova-local** local volume group. For more information, see *Managing Physical Volumes* on page 129.

In addition, you must set the instances logical volume size. For more information, see *Instances Logical Volume Considerations* on page 69.

Managing Physical Volumes

You can add, delete, and review physical volumes on a compute host.

Physical volumes provide storage using local disks. You can use the web administration or the CLI to manage them. For web administration interface instructions, see *Configuring a Compute Host to Provide Local Storage* on page 64.

As each physical volume is created, it is added to an existing local volume group.

Before you can modify the settings for a host, you must lock the host:

```
~(keystone_admin)$ host-lock hostname
```

Before you can add a physical volume, a local volume group must exist on the host. To create one, see *Managing Local Volume Groups* on page 128

The following CLI commands are available for managing physical volumes.

```
~(keystone_admin)$ host-pv-add hostname groupname disk_uuid
```

where **groupname** is the name of the local volume group to which the physical volume is added.

Note:

The only valid **groupname** is **nova-local**.

When **disk_uuid** indicates the root disk, the physical volume uses a system-designated partition on the root disk. For any other disk, the physical volume uses the entire disk.

```
~(keystone_admin)$ host-pv-delete hostname nova-local disk_uuid
```

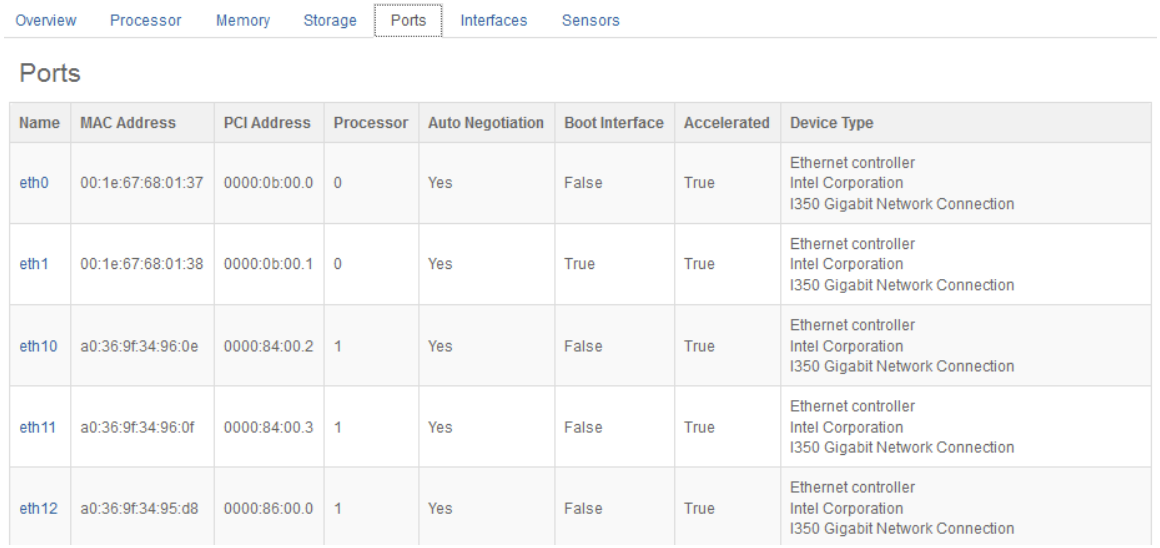
```
~(keystone_admin)$ host-pv-list hostname
```

```
~(keystone_admin)$ host-pv-show hostname
```

To configure VMs to use local storage, see *Specifying Local Storage for VM Ephemeral Resources* on page 153.

Ports

The tab **Ports** on the **Inventory Detail** page presents information about the physical ports on a host, as illustrated below.



Currently none of the port attributes is configurable; they are all read directly from the hardware. Port information is presented in several columns, as follows:

Name

The name of the physical port, as identified by the host's Linux kernel.

MAC Address

The port's unique MAC address.

PCI Address

The port's unique address on the PCI bus. Together with the MAC address, this field can be used to uniquely identify a port on the host's hardware platform.

Processor

The processor node that the port's IO controller is connected to.

Auto Negotiation

The status of the Ethernet auto-negotiation flag. Currently, auto-negotiation is always enabled.

Boot Interface

The boot flag, whether or not PXE booting is enabled.

Accelerated

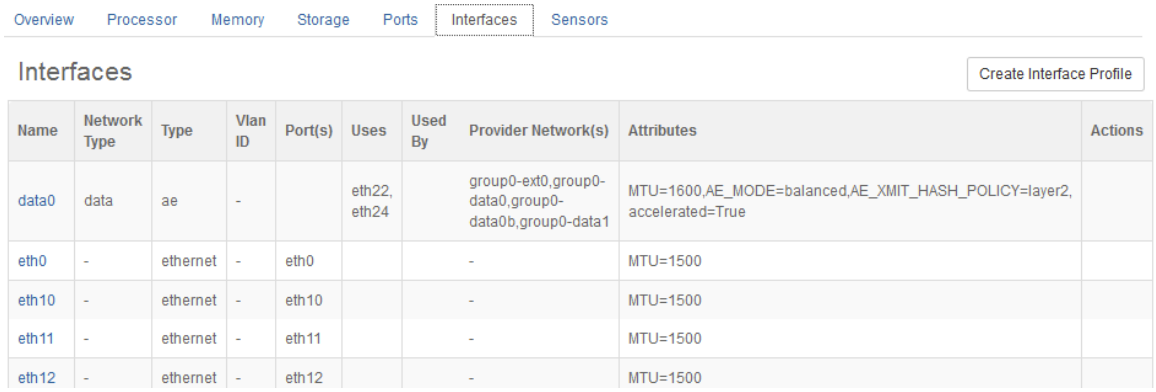
The acceleration status. If the port is supported by AVS using DPDK poll-mode drivers, acceleration will be used. Otherwise, it will operate in non-accelerated mode with AVS leveraging kernel drivers.

Device Type

Hardware information about the port type, manufacturer, and model.

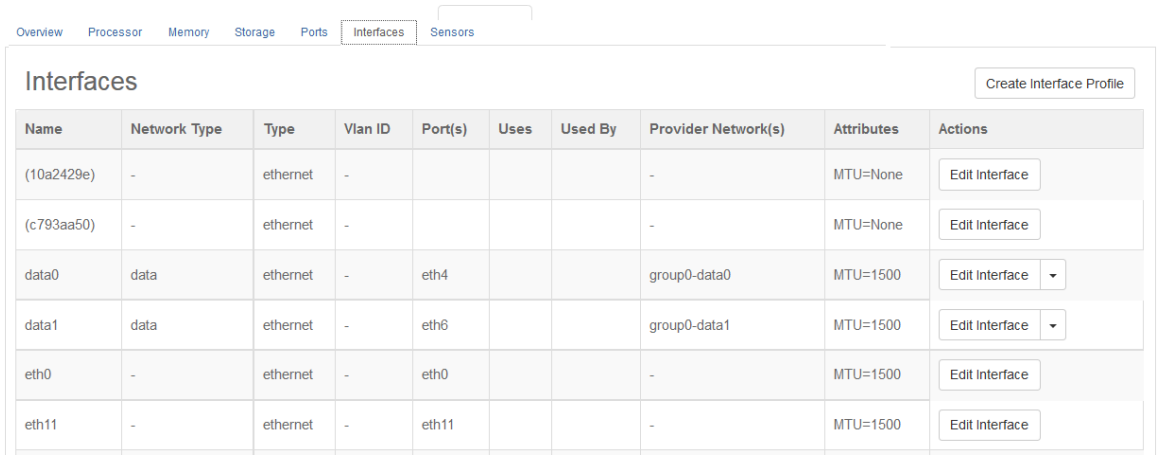
Interfaces

The **Interfaces** tab on the **Inventory Detail** page presents details about the logical L2 network interfaces on a node, as illustrated below for an unlocked controller node.



In this example the node has two allocated interfaces, **eth0** connecting to the OAM network, and **eth1** connecting to the internal management network. These interfaces are mandatory and therefore cannot be deleted. They reflect the allocation given when the node was provisioned. See the *HP Helion OpenStack Carrier Grade Software Installation Guide* for details.

On a properly configured compute node, the **Interfaces** tab presents additional logical interfaces, as illustrated below for a locked node:



In this example the node has three allocated interfaces, **eth1** connecting to the internal management network, and **data0** and **data1** connecting to the provider networks **group0-data0** and **group0-data1** respectively. The interface **eth1** is auto-provisioned as part of the automated software installation process on the compute node. It can be modified but not deleted.

Information about interfaces is presented in several columns, as follows:

Name

The name given to the logical L2 interface.

Network Type

The type of network the logical network interface is connected to. The options are:

- **data**, for a compute node data interface
- **infra**, for the optional infrastructure network
- **mgmt**, for the internal management network
- **oam**, for the OAM network
- **pci-passthrough**, for a PCI passthrough interface

Type

Ethernet, or aggregated Ethernet (LAG).

Vlan ID

The VLAN ID of the network listed in the **Network Type** column, if the network uses a shared interface. For more information about shared interfaces, see *Shared (VLAN) Ethernet Interfaces* on page 24.

Port(s)

The physical ports on top of which the logical interface is built. Multiple ports are displayed when the logical interface uses LAG.

Uses

The interface used by the network listed in the **Network Type** column, if the network uses a shared interface. The VLAN ID of the network is shown in the **Vlan ID** field.

Used By

The networks that share the interface using VLAN tagging, if the interface is shared. For more information about shared interfaces, see *Shared (VLAN) Ethernet Interfaces* on page 24.

Provider Networks

This option is relevant for compute nodes only, and for interfaces of the **data** network type. It lists the provider networks associated with the data interface.

Attributes

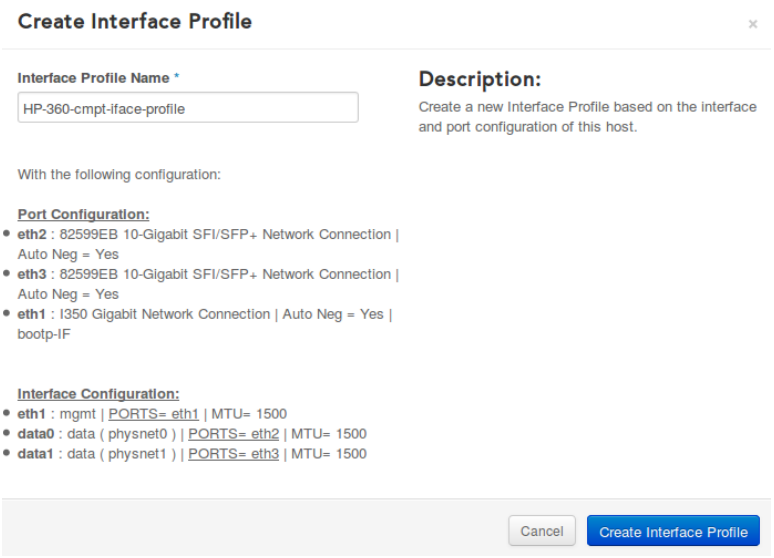
Details including the current MTU size for the interface and whether the interface is DPDK-accelerated

Actions

On a locked node, you can modify a logical interface, and execute management operations on it. This is implemented using the buttons **Edit Interface** and **More**. These buttons are not available when the node is unlocked.

Creating an Interface Profile

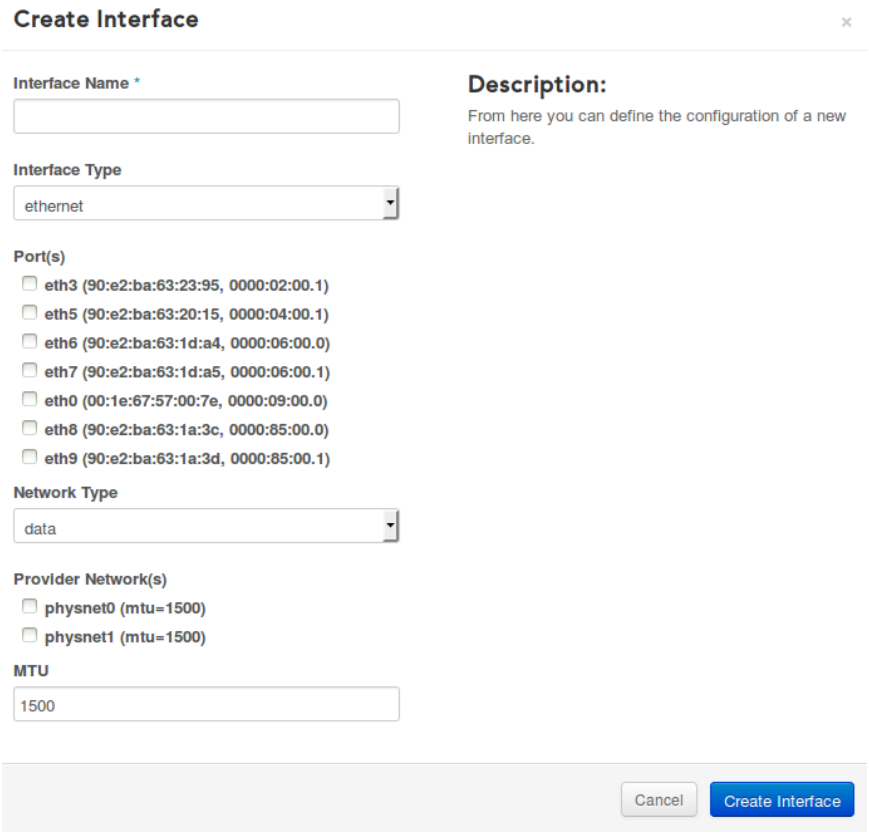
Use the **Create Interface Profile** button on the **Interfaces** tab to display the **Create Interface Profile** window. From there you can create a new interface profile out of the currently defined node interfaces. An example for a compute node is illustrated below.



In this window you enter the name of the new profile, **HP-360-cmpt-iface-profile** in this case, and then click the button **Create Interface Profile** to execute. Available interface profiles are listed, and can be removed, from the **System Inventory** page, as described in *Hardware Profiles* on page 110.

Managing Interfaces

New logical interfaces can be created using the **Create Interface** button on the **Interfaces** tab. This button is available only on nodes in the locked state. When the button is clicked, the window **Create Interface** is displayed, as illustrated below.



The following fields are available:

Interface Name

The name to be assigned to the new logical interface.

Interface Type

The type of network interface, Ethernet or aggregated Ethernet (LAG).

Ports

A list of physical ports available to the new logical interface, with corresponding check-mark boxes. Physical ports already allocated to other interfaces are not listed.

Network Type

The type of network to attach the new logical interface to (**data** or **infra**, for compute node data interfaces and infrastructure network connections respectively). Note that connections to the OAM and internal management networks are auto-provisioned during the software installation process.

Provider Network(s)

A list of provider networks available to the new logical interface, with corresponding check-mark boxes. They can be selected for networks of type **data**, to which multiple provider networks can be attached. Provider networks already allocated to other data interfaces are not listed, since no provider network can be associated with more than a single data interface.

MTU

The MTU size in bytes for the interface. For compute nodes, values between 1500 and 9000 are supported. For controller nodes, the MTU size cannot exceed 1500.

To edit or remove a logical interface of network type **data** or **infra** you must first lock the target node. You can then use the **Edit Interface** and **More** buttons to perform these maintenance actions.

Logical interfaces of network types **oam** and **mgmt** cannot be deleted. They can only be modified to use different physical ports when required.

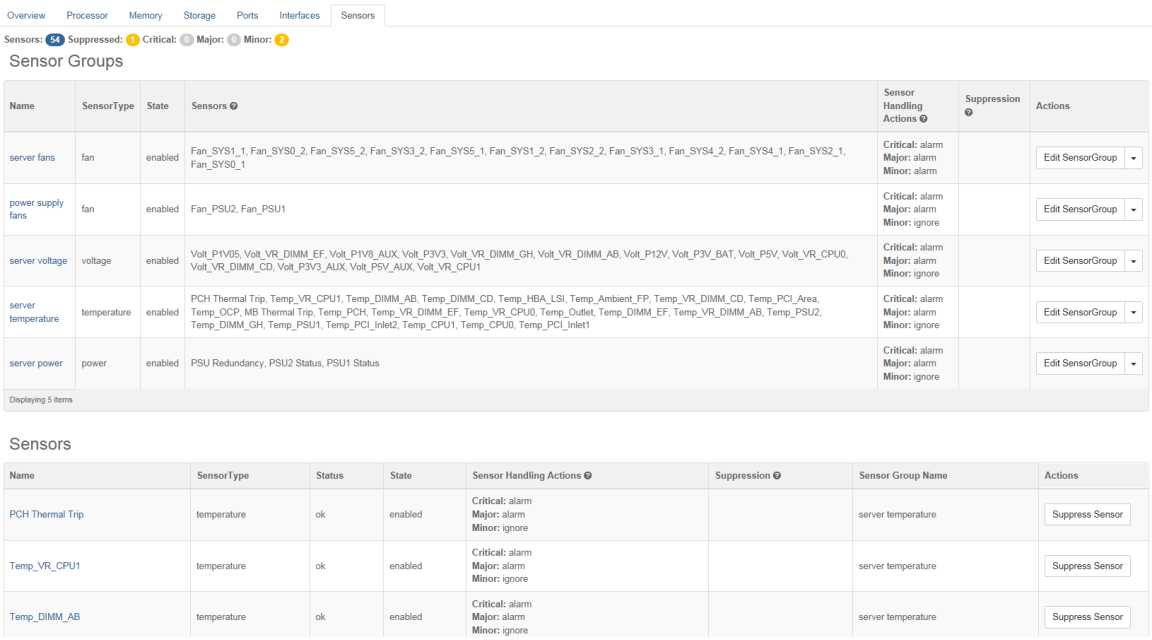
Sensors

The **Sensors** tab on the **Inventory Detail** page presents details for sensors used to monitor host hardware health.

If an optional board management control (BMC) module is present on the host and configured for the host in the HP Helion OpenStack Carrier Grade inventory, and sensor support is implemented for the BMC type in HP Helion OpenStack Carrier Grade, this tab lists the available sensors and shows their status. It also lists *sensor groups* that have been defined for the sensors.

Note:

Currrently, sensor support is implemented for Quanta BMCs. For more about BMC modules, see *The Board Management Network* on page 39.



Sensor Status

Each of the individual sensors in the **Sensors** list can report the following **Status** conditions:

- OK
- Minor
- Major
- Critical

The status of each sensor is audited periodically to refresh the system monitoring data. You can configure the refresh period, or **audit interval**, for sensor groups.

Actions

You can configure different actions for each status level using sensor groups. The configured action applies to all sensors in the sensor group. Currently two types of action are supported:

alarm

This generates a HP Helion OpenStack Carrier Grade alarm with a severity level corresponding to the sensor status level. For more about HP Helion OpenStack Carrier Grade alarms, see *Fault Management* on page 170.

For the status levels **Major** and **Critical**, it also sets the **Availability State** of the host to **Degraded**. For more about availability states, see *Host Inventory* on page 103.

ignore

The status is reported in the Sensor list, but no action is taken.

You can suppress the configured **Action** for individual sensors or groups of sensors. Suppressed sensors are still audited, and their status is reported in the **Sensors** list. For more information, see *Suppressing Sensor Actions* on page 137.

Sensor Groups

Sensors that perform the same type of monitoring are collected into **Sensor Groups**. You can configure audit intervals and alarm actions for each group. The configured values apply to all sensors in the group. For more information, see *Adjusting Sensor Actions and Audit Intervals* on page 136.

The available Sensor Groups and their membership are predefined.

server fans

Sensors that monitor the speed and health of CPU cooling fans.

power supply fans

Sensors that monitor the speed and health of power supply cooling fans.

server voltage

Sensors that monitor DC voltage levels supplied to components.

server temperature

Sensors that monitor component or ambient temperatures.

server power

Sensors that monitor the operational status of power supplies.

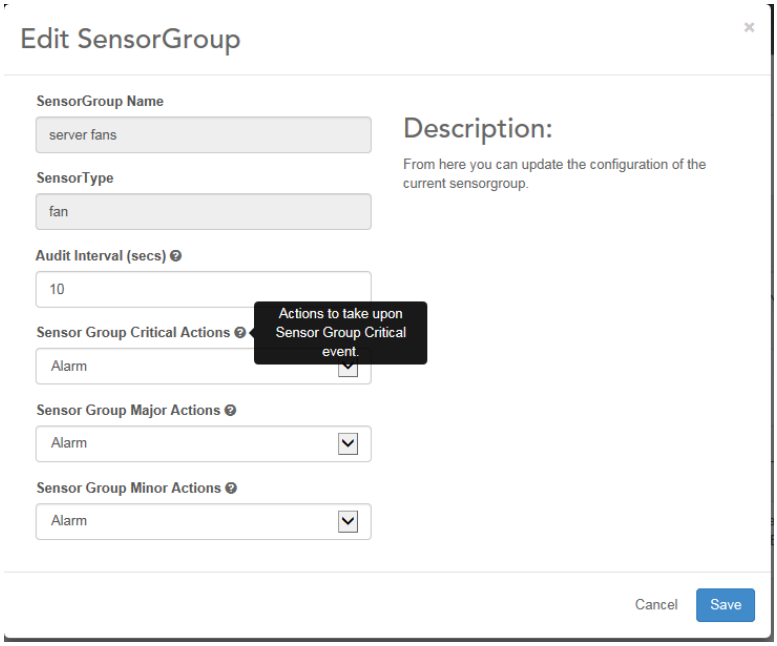
Adjusting Sensor Actions and Audit Intervals

You can configure audit intervals and actions for groups of sensors.

For more information about sensors and sensor groups, see *Sensors* on page 134.

To use the command-line interface, see *CLI Commands for Managing Sensors* on page 137.

1. Open the **Inventory Detail** page for the host.
   a) On the **Admin** menu of the web administration interface, open the **System Panel** section, and then select **Inventory**.
   b) Select the **Hosts** tab, and then in the **Host Name** column, click the name of the host.
2. Select the **Sensors** tab.
3. In the **Sensor Groups** list, click **Edit SensorGroup** for the group you want to configure.



In the **Edit SensorGroup** dialog box, change the settings as required.

Audit Interval

The time, in seconds, to wait between sensor audits. At each audit, the sensor status reading is refreshed. Changes to the audit interval do not take effect until the current interval expires.

Sensor Group Critical Actions

The action to take if the sensor status is **Critical**. If this is set to **Alarm**, then when this status is reported, a corresponding HP Helion OpenStack Carrier Grade alarm is generated, and the host availability is set to **Degraded**.

Sensor Group Major Actions

The action to take if the sensor status is **Major**. If this is set to **Alarm**, then when this status is reported, a corresponding HP Helion OpenStack Carrier Grade alarm is generated, and the host availability is set to **Degraded**.

Sensor Group Minor Actions

The action to take if the sensor status is **Minor**. If this is set to **Alarm**, then when this status is reported, a corresponding HP Helion OpenStack Carrier Grade alarm is generated.

Suppressing Sensor Actions

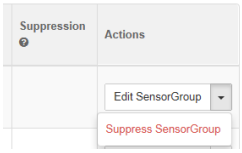
You can suppress the configured **Action** for individual sensors or groups of sensors

If a sensor is faulty, or is generating frequent minor alarms for a known condition that cannot be addressed immediately, you can prevent it from generating further alarms. Suppressed sensors are still audited, and their status is reported in the **Sensors** list.

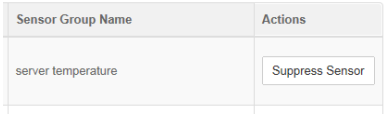
For more information about sensors and sensor groups, see *Sensors* on page 134.

To use the command-line interface, see *CLI Commands for Managing Sensors* on page 137.

1. Open the **Inventory Detail** page for the host.
   a) On the **Admin** menu of the web administration interface, open the **System Panel** section, and then select **Inventory**.
   b) Select the **Hosts** tab, and then in the **Host Name** column, click the name of the host.
2. Select the **Sensors** tab.
3. Use the controls on the **Sensors** tab to suppress actions for individual sensors or sensor groups.
   - To suppress actions for a group of sensors, open the **Actions** menu for the group, and then select **Suppress SensorGroup**.



   - To suppress actions for an individual sensor, locate the sensor in the **Sensors** list, and then click **Suppress Sensor**.



The **Suppression** field in the list is updated to show that actions are suppressed for the sensor.

CLI Commands for Managing Sensors

You can use the command-line interface to list sensor information and change sensor settings.

The following CLI commands are available for working with sensors. For complete syntax information, refer to the command help.

- `system host-sensor-list`
- `system host-sensor-modify`

You can modify sensors using the suppress parameter (True or False).

- `system host-sensor-show`
- `system host-sensorgroup-list`
- `system host-sensorgroup-modify`

  You can modify sensor groups using the following parameters:

  - actions_critical_group (valid values are alarm or ignore)
  - actions_major_group (valid values are alarm or ignore)
  - actions_minor_group (valid values are alarm or ignore)
  - audit interval_group (time in seconds)
  - suppress (True or False)
- `system host-sensorgroup-show`

For example:`~(keystone_admin)$ system host-sensor-modify controller-0 \ d9af9433-44dd-4526-b0fd-8d7a0cdb877b suppress=True`

For more information about sensors and sensor groups, see *Sensors* on page 134.

Controller Nodes and High Availability

Services in the controller nodes run constantly in active/standby mode to provide continuity in the event of a controller failure.

Controller services are organized internally into the following groups:

Table 5: Controller Service Groups

Group	Description
Cloud Services	The enhanced OpenStack components, including Nova, Neutron, Cinder, Ceilometer, and Heat
Controller Services	Core HP Helion OpenStack Carrier Grade services such as maintenance and inventory
Directory Services	LDAP services
OAM Services	OAM access services
Patching Services	Patching alarm services
Storage Monitoring Services	Storage alarm services
Storage Services	Storage REST API services
Web Services	The HP Helion OpenStack Carrier Grade OpenStack Horizon service and web server

Each of these groups is run in 1:1 HA mode by the controllers. This means that while some service groups can be active on controller-0, and in standby on controller-1, others are active on controller-1, and in standby on controller-0.

The high-availability framework constantly monitors and reports on the health of the individual services within each of the service groups on both controllers. When a service fails, a decision is made on whether to restart it on the same controller, or to switch the corresponding service group to the other controller. This decision depends on the criticality and the dependencies of the affected service.

For maintenance purposes, when one of the controller nodes needs to be powered down for service, it is necessary to force all currently active service groups in one controller to switch to the other. This can be done from the **Hosts** tab on the **Inventory** page, by selecting the option swact (switch active) in the **More** menu of the controller you want to take out of service.

The Active Controller

Services in the **Controller Services** group drive core functionality in HP Helion OpenStack Carrier Grade. The controller where they are running is referred to as the *active* controller. The **Hosts** tab in the **System Inventory** page of the **Admin** panel lists the status of all hosts in the cluster; it reports the active controller as having the *Controller-Active* personality.

When working from the CLI on a controller node it is often important to ensure that you are working on the active controller, for example, to execute OpenStack **admin** operations, or to change the password of the **wrsroot** user account. See *Linux User Accounts* on page 18 for further details on the **wrsroot** account.

You can ensure you are working on the active controller by using the OAM floating IP address as the destination address in the SSH command.

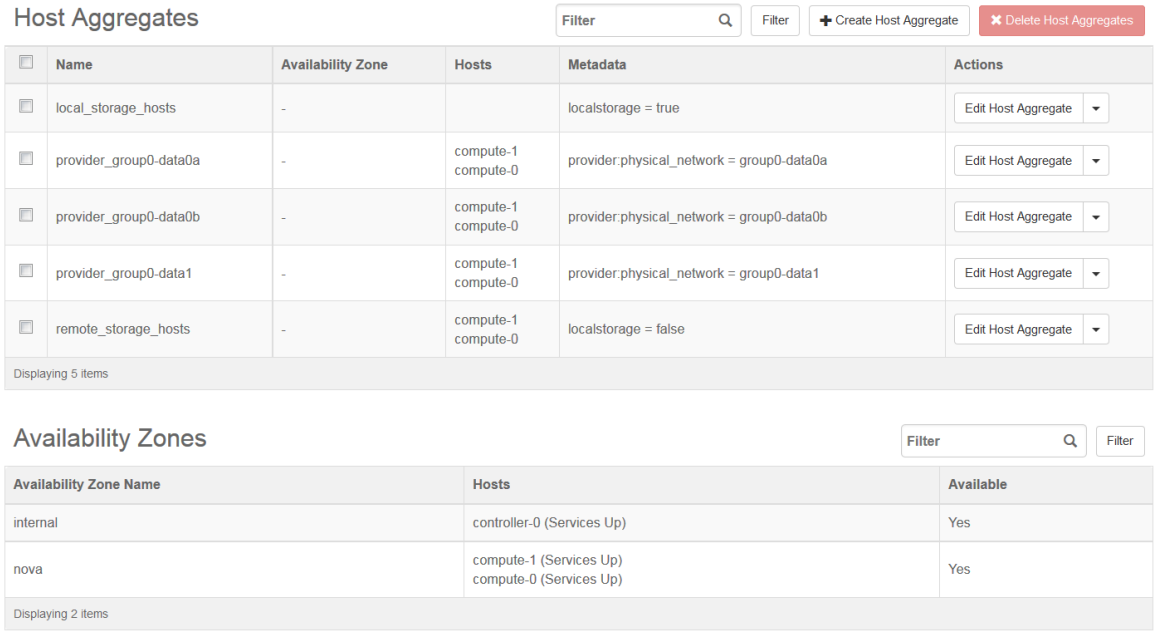
Host Aggregates

Host aggregates are collections of hosts that share common attributes for the purposes of VM scheduling.

Note:

The information in this topic is preliminary and subject to change.

To view host aggregates, open the **Host Aggregates** page from the **Admin** menu.



When the Nova Scheduler selects a compute node to instantiate a VM, it can use host aggregates to narrow the selection. For example, if the VM requires local storage, the Nova scheduler selects a host from the **local_storage_hosts** host aggregate. Alternatively, if the VM requires remote storage, the scheduler selects from the **remote_storage_hosts** host aggregate. This ensures that the instance is instantiated on a host that meets the requirements of the VM.

The Nova scheduler does not always use host aggregates. For example, if a VM does not specify either local or remote storage, the Nova scheduler can instantiate it on any resource.

Some host aggregates are managed automatically by HP Helion OpenStack Carrier Grade.

Caution:

Do not make manual changes to host aggregates that are managed automatically.

- The **local_storage_hosts** and **remote_storage_hosts** memberships are updated automatically whenever a local volume group is added or removed on a compute host. For more information, see *Configuring a Compute Host to Provide Local Storage* on page 64.

You can use host aggregates to meet special requirements. For example, you can create a pool of compute hosts to offer dedicated resources such as pinned NUMA nodes or specific huge page sizes, while grouping the remaining compute hosts to offer shared resources.

Chapter 5

Managing Virtual Machines

Topics:

Virtual machines and the applications that run on them are at the core of cloud-based communications solutions. Understanding how to manage them is of paramount importance.

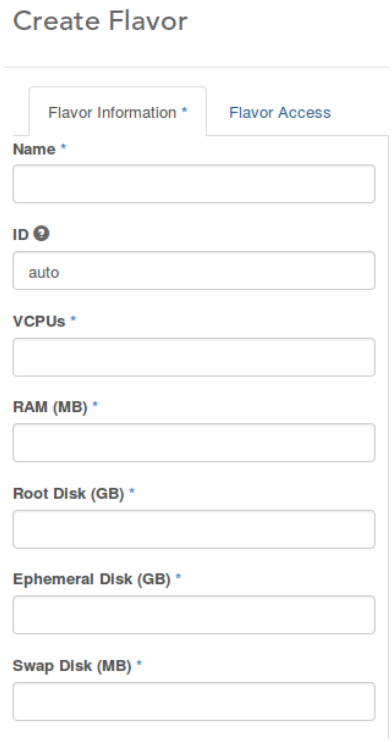
This chapter elaborates on software features available exclusively on HP Helion OpenStack Carrier Grade, which add to the already powerful set of management tools available from OpenStack. The exclusive features include virtual machine settings to optimize the use of virtual CPUs, improvements to the scheduling algorithms, better integration of NUMA architectures, and others.

Virtual Machine Flavors

A flavor is a list of attributes applied to a virtual machine when a new instance is created. It specifies resources to be allocated by the system, such as size of RAM, number of cores, storage resources, and so on.

You can create and edit flavors using the **Flavors** page of the web administration interface. To access this page, open the **Admin** menu, expand the **System** section, and then select **Flavors**.

- To create a flavor, click the **Create Flavor** button. The **Create Flavor** window appears, as illustrated below:



The **Create Flavor** window has two tabs.

Flavor Information

Defines basic information for the flavor.

Name

The name to be associated with the flavor.

ID

The object ID to associate with the flavor. In most situations you should leave the default value of **auto** for the system to auto-generate a unique ID.

VCPUs

Number of virtual CPUs to be allocated to the virtual machine.

RAM MB

Memory, in megabytes, to be allocated to the virtual machine.

Root Disk GB

Specifies the virtual root disk size in gigabytes. This is an ephemeral disk to which the base image is copied. Use the value **0** to set the ephemeral disk size equal to the base image size. This value is ignored when booting from a Cinder volume.

Note:

Booting the VM from an ephemeral root disk is supported only if local storage is enabled for the flavor. For more information, see *Specifying Local Storage for VM Ephemeral Resources* on page 153.

Ephemeral Disk GB

Specifies the size, in gigabytes, of a secondary ephemeral data disk. This is an empty, unformatted disk that exists only for the life time of the instance.

Note:

To use non-zero ephemeral disk space, you must enable local storage for the flavor. For more information, see *Specifying Local Storage for VM Ephemeral Resources* on page 153.

Swap Disk MB

Ephemeral swap space, in megabytes, to be allocated to the instance.

Note:

To use non-zero ephemeral disk space, you must enable local storage for the flavor. For more information, see *Specifying Local Storage for VM Ephemeral Resources* on page 153.

Flavor Access

Controls which projects can see and use the flavor.

- To edit an existing flavor, click its name in the **Flavor Name** column. For more information about editing flavors, see *Flavor Extra Specifications* on page 143.

Flavor Extra Specifications

You can edit an existing flavor to include additional attributes using *extra specifications*.

Extra specifications are key-value pairs you can add to an existing flavor to be included when the flavor is used with a new virtual machine. The HP Helion OpenStack Carrier Grade **Extra Specs** tab for a flavor includes extensions specific to HP Helion OpenStack Carrier Grade.

To access the extra specifications settings for a flavor, open the **Flavors** window and click the flavor name. For help accessing the Flavors window, see *Virtual Machine Flavors* on page 142.

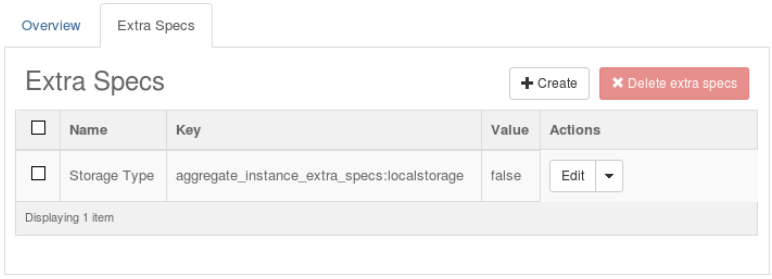


Figure 5: Flavor extra specs

The **Extra Specs** tab lists extra specifications that have been added for the flavor.

Note:

A **Storage Type** extra spec is added by default for all new flavors. To ensure that instantiated VMs use storage of a known type, do not delete this extra spec. For more information, see *Specifying Local Storage for VM Ephemeral Resources* on page 153.

To modify an existing entry, use the **Edit** button. To remove a single entry, open the drop-down menu associated with the extra spec, and select **Delete extra spec** from the menu. You can also remove all extra specs at once using the **Delete extra specs** button.

To add a new extra specification, click **Create**. This opens the **Create Flavor Extra Spec** window, as illustrated below:

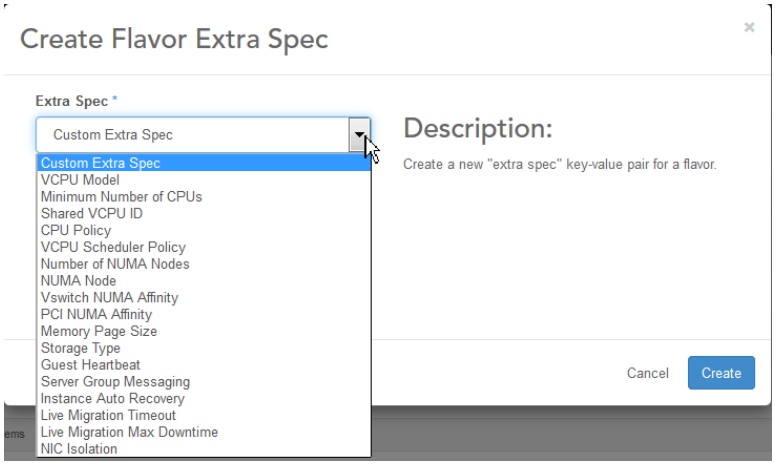


Figure 6: Creating a flavor extra specification

Use the **Extra Spec** drop-down list to add specifications. The following options are available:

Custom Extra Spec

Available for internal use only.

VCPU Model

The CPU model to use with the virtual machine. If this extra spec is not added, then by default the QEMU processor is used. For more information, see *Specifying the VCPU Model for a VM* on page 146.

Minimum Number of CPUs

Sets the minimum number of virtual CPUs for the flavor. If this extra spec is not added, then by default the minimum is one. For more information, see *Setting the CPU Scaling Range* on page 163.

Shared VCPU ID

The ID of a virtual CPU scheduled to run on a shared physical CPU in the compute host. If this extra spec is not added, then by default the VM does not use a shared CPU. For more information, see *Designating Shared Physical CPUs on a Compute Host* on page 120 and *Pinning a vCPU to a Shared Physical CPU* on page 146.

Note:

To use this extra specification, you must also set the **CPU Policy** extra specification for the flavor to Dedicated. This enables the VM to run a single house-keeping virtual CPU on a shared physical core, while keeping all its high-performance virtual CPUs on dedicated physical cores.

CPU Policy

The policy for assigning dedicated physical or logical CPU resources to the VM. If this extra spec is not added, then by default resources are drawn from a shared pool of cores. For more information, see *Specifying Dedicated CPUs for a VM* on page 147.

VCPU Scheduler Policy

Sets the scheduling priority for non-boot virtual CPUs. If this extra spec is not added, then by default *nice* priority 0 is assigned. For more information, see *Configuring vCPU Scheduling and Priority* on page 148.

Note:

This extra spec is shown only for flavors with more than one CPU.

Number of NUMA Nodes

Sets the number of virtual NUMA Nodes. If this extra spec is not added, then by default the memory and CPU resources for a flavor are assigned to a single virtual NUMA Node. For more information, see *Configuring the NUMA Node Allocations for a VM* on page 149.

NUMA Node

The NUMA node to use when launching a virtual machine. If this extra spec is not added, then by default any available NUMA node is used, subject to other NUMA node settings. For more information, see *Pinning a Guest NUMA Node to a Host NUMA Node* on page 151.

Vswitch NUMA Affinity

Specifies that a NUMA node with a vSwitch core is required or preferred when instantiating the VM. If this extra spec is not added, then by default any available NUMA node is used, subject to other NUMA node settings. For more information, see *Affining a VM to a NUMA Node with a vSwitch Core* on page 151.

PCI NUMA Affinity

For a VM configured to use an SR-IOV or PCI passthrough interface, specifies that a NUMA node with direct access to the physical interface is either required or preferred for instantiation. If this extra spec is not added, then by default a NUMA node with direct access is used. For more information, see *Specifying Best Effort for PCI NUMA Node Affinity* on page 152.

Memory Page Size

Sets the page size for VM memory. If this extra spec is not added, then by default small pages are used. For more information, see *Specifying a Page Size for a VM* on page 152.

Storage Type

Specifies whether to use local or remote ephemeral storage resources. This extra spec is added automatically, and is set for remote storage by default. For more information, see *Specifying Local Storage for VM Ephemeral Resources* on page 153.

Guest Heartbeat

Enables the Heartbeat API for use by guests on the VM. If this extra spec is not added, then by default this API is disabled. For more information, see *Enabling the Guest Heartbeat API for a VM* on page 154.

Server Group Messaging

Enables the VM to communicate with other VMs in the same server group using a messaging API. If this extra spec is not added, then by default this API is disabled. For more information, see *Enabling Server Group Messaging for a VM* on page 155.

Instance Auto Recovery

Disables auto recovery of failed virtual machine instances. If this extra spec is not added, then the system will automatically recover failed virtual machine instances. For more information, see *Disabling Instance Auto Recovery* on page 155.

Live Migration Timeout

Specifies the number of seconds to wait for a live migration to complete. If this extra spec is not added, then it defaults to 800 seconds. For more information, see *Live Migration of Virtual Machines* on page 161 and *Specifying a Live Migration Timeout* on page 156.

Live Migration Max Downtime

Specifies the number of milliseconds of downtime to accept during live migrations. If this extra spec is not set, then it defaults to 500 milliseconds. For more information, see *Specifying Live Migration Maximum Downtime* on page 157.

NIC Isolation

Enables isolating a virtual machine instance's network interface cards from all other interfaces running on the host. If this extra spec is not set, then the instance's interfaces are not isolated. For more information see *Isolating an Untrusted Guest* on page 157.

Specifying the VCPU Model for a VM

You can select a particular VCPU model for a VM in order to leverage advanced CPU features such as SSE4.2, AES, or AVX on the compute nodes.

When a virtual machine is launched, a Nova scheduler filter restricts the target compute nodes to those with available cores of the requested model or better. If no such compute node is available, the error *No valid host was found* is reported.

If no **VCPU Model** extra spec is added, then by default the QEMU virtual processor is used.

The following selections are available:

- Intel Core i7 9xx (Nehalem Class Core i7)

  (This is offered by default when the extra spec is added using the web administration interface.)
- Intel Westmere E56xx/L56xx/X56xx (Nehalem-C)
- Intel Xeon E312xx (Sandy Bridge)
- Intel Core Processor (Haswell)

Note:

For some selections, limitations apply:

- The default QEMU processor is based on the Intel Core Duo CPU, which does not support extended instruction sets such as SSSE3 or SSSE4.2 used for DPDK networking or other advanced features.
- The Haswell model does not currently support transactional synchronization extensions (TSX).

To add this extra spec to a flavor using the web administration interface, use the **VCPU Model** selection in the **Create Flavor Extra Spec** drop-down menu. To access this menu, see *Flavor Extra Specifications* on page 143.

You can also specify the extra spec from the CLI using the following command:

```
~(keystone_admin)$ nova flavor-key flavor_name set hw:cpu_model=cpu_model
```

where *cpu_model* is one of the following:

- Nehalem
- Westmere
- SandyBridge
- Haswell

Note:

If any other value is supplied, it is ignored and the default QEMU model is used.

If the hw:cpu_model parameter is not supplied with the `nova flavor-key` command, then the default QEMU model is used.

Pinning a vCPU to a Shared Physical CPU

You can pin a vCPU to a shared physical CPU by using a flavor with the required extra specification.

You can set up a shared physical CPU on the host to run low-load or non-real-time tasks for multiple VMs, freeing other cores on the host for dedicated high-load tasks. You can then use an extra spec to pin a specified vCPU to the shared physical CPU.

To add the required extra specification to a flavor using the web administration interface, use the **Shared VCPU ID** selection in the **Create Flavor Extra Spec** drop-down menu. To access this menu, see *Flavor Extra Specifications* on page 143.

To set the Shared VCPU ID from the CLI, use a command of the following form:

```
~(keystone_admin)$ nova flavor-key flavor_name set
 hw:wrs:shared_vcpu=vcpu_id
```

where **vcpu_id** is an integer that identifies the vcpu. The valid range starts at 0 and must be less than the **VCPUs** value defined for the flavor.

To use this extra spec, you must define a shared physical CPU on at least one host. For more information, see *Designating Shared Physical CPUs on a Compute Host* on page 120. To support migration, shared physical CPUs on multiple hosts are recommended.

You must also set the **CPU Policy** extra specification for the flavor to **Dedicated**. This enables the high-load vCPUs for the instance to be pinned.

Specifying Dedicated CPUs for a VM

You can require the use of dedicated CPUs for a VM using an extra specification.

When a virtual machine is launched, its virtual CPUs use resources from a shared pool of cores on the target compute node. Each virtual CPU is scheduled for processing as an independent thread. The CPU Policy extra specification involves two aspects of the scheduling process: the CPU affinity mask (that is, the conditions required for a target CPU to be eligible), and the CPU over-commit limit (an independent consideration that affects whether an otherwise eligible target CPU may be used.) The extra spec controls whether the virtual CPU thread can share a core with other vCPU threads.

Until this extra spec is explicitly defined, the effective default is **Shared** to permit sharing, subject to the over-commit limit. You can change this by adding the extra spec. When you do so using the web administration interface, the non-default policy **Dedicated** is offered automatically for convenience.

When the CPU Policy is set to **Dedicated**, each virtual CPU thread is scheduled to run on a single core exclusively. The cores are removed from the shared resource pool, and each virtual CPU is scheduled to run as a dedicated thread on its corresponding target core. The cores are returned to the shared pool upon termination of the virtual machine.

Specifying **Dedicated** effectively affines each virtual CPU thread to a dedicated core, and ensures that the core is not shared with other virtual CPU threads. The **Dedicated** setting is recommended for Carrier-Grade requirements to prevent over-commitment of host resources, and to minimize the impact that scheduling events may have on the latency and throughput responses of the guest kernel.

Note:

The **Dedicated** setting is also required for CPU scaling.

When the CPU Policy is **Shared**, the virtual CPU threads are scheduled to run on any available core on the compute node. Also, they can be moved within the compute node from one core to another at any time. In a sense, the virtual CPU threads can be considered to be floating threads. This is the CPU affinity mask portion of the scheduling action.

Additionally, the virtual CPU threads can share the core with other threads from the same virtual machine or other virtual machines. The default OpenStack virtual CPU over-commit ratio is 16:1, which means that up to 16 virtual CPU threads can share a single core. Over-committing of virtual CPUs is highly discouraged in Carrier-Grade operations.

To add this extra spec to a flavor using the web administration interface, use the **CPU Policy** selection in the **Create Flavor Extra Spec** drop-down menu. To access this menu, see *Flavor Extra Specifications* on page 143.

To add the extra spec using the CLI, use the following command:

```
~(keystone_admin)$ nova flavor-key flavor_name set hw:cpu_policy=policy
```

where *policy* is either dedicated or shared. If any other value is suppied, the policy is set to **Dedicated**.

Note:

Using a single compute node for both shared and dedicated threads is not recommended. For clusters that must support threads of both types, use a host aggregate for each type. For more information, see *Host Aggregates* on page 139.

Configuring vCPU Scheduling and Priority

You can assign the Linux scheduler and priority for non-boot virtual CPUs using an extra specification.

This extra specification applies to non-boot virtual CPUs. For the boot CPU, the Linux scheduler and priority are fixed to normal time-shared scheduling policy with a *nice* priority of 0.

Note:

This extra spec is shown only for flavors with more than one CPU.

For each additional virtual CPU, the available options are:

Default Policy

Assigns a normal time-shared scheduling policy with *nice* priority of 0.

Real-Time FIFO

Assigns a real-time, first-in-first-out policy, with priority in the range of 1–99, specified in the associated **VCPU Priority** field.

Real-Time Round Robin

Assigns a real-time, round-robin policy, with priority of 1–99, specified in the associated **VCPU Priority** field.

Note:

The scheduling policy and priority level set by this extra spec become effective when the virtual CPU thread is scheduled to run on regularly shared cores. Virtual CPU threads pinned to run on shared-only CPUs always use the default policy of time-shared scheduling with *nice* priority of 0. See *Pinning a vCPU to a Shared Physical CPU* on page 146 for details.

To add this extra spec to a flavor using the web administration interface, use the **VCPU Scheduler Policy** selection in the **Create Flavor Extra Spec** drop-down menu. To access this menu, see *Flavor Extra Specifications* on page 143.

You can specify the scheduler policy and priority for non-boot virtual CPUs from the CLI using the hw:wrs:vcpu:scheduler parameter on the `nova flavor-key` command. This parameter accepts a semicolon-separated list of *scheduler:priority:vcpus* values enclosed by quotes, as follows.

scheduler

The scheduler policy. One of *other*, *fifo*, or *rr* to indicate normal time-shared, FIFO, and Round Robin policies respectively.

priority

The real-time scheduler priority. A value between 1 and 99.

vcpus

An optional list of virtual CPUs as a comma-separated list (1,2,3), or a range specification (1-3). Virtual CPU number 0 refers to the boot virtual CPU and therefore cannot be used.

To set real-time schedulers and priorities on virtual CPU 1 (FIFO, 50) and virtual CPU 2 (Round Robin, 80):

```
~(keystone_admin)$ nova flavor-key flavor_name set
 hw:wrs:vcpu:scheduler="fifo:50:1;rr:80:2"
```

To set real-time Round Robin schedulers for three virtual CPU (Round Robin, 80):

```
~(keystone_admin)$ nova flavor-key flavor_name set
 hw:wrs:vcpu:scheduler=rr:80:1-3
```

To set the FIFO scheduler with priority 50 on all virtual CPUs except virtual CPU 0:

```
~(keystone_admin)$ nova flavor-key flavor_name set
 hw:wrs:vcpu:scheduler=fifo:50
```

To reset all scheduler settings to default values (non real-time scheduler with priority 0):

```
~(keystone_admin)$ nova flavor-key flavor_name unset
 hw:wrs:vcpu:scheduler
```

Configuring the NUMA Node Allocations for a VM

You can use flavor extra specs or image properties to allocate virtual memory and vCPU resources to virtual NUMA nodes.

By default, the memory and CPU resources defined for a flavor are assigned to a single virtual NUMA node, which is mapped to an available host NUMA node when an instance is launched or migrated. You can specify the use of multiple NUMA nodes using either flavor extra specifications (which take the general form **hw:specification**), or image properties (which take the general form **hw_specification**).

By default, the available CPU and memory resources for the flavor are distributed equally among the NUMA nodes. You can customize the distribution of resources. For example, given a flavor with two NUMA nodes, three vCPUs, and 1024 MB of RAM, you can assign one vCPU and 512 MB to virtual NUMA node 0, and two VCPUs and 512 MB to virtual NUMA node 1. The ability to allocate resources is useful when pinning virtual NUMA nodes to host NUMA nodes to optimize VM performance.

By default, the memory required for a virtual NUMA node is pinned to a single host NUMA node to ensure high performance. For applications where this is not a concern, you can relax the memory allocation requirements so that the memory for a virtual NUMA node can be drawn from more than one host NUMA node if necessary.

When deploying network guest images operating on the data path, it is advisable to co-locate the virtual machines, the AVS switch, physical ports, and all other networking elements on the same node.

Use of this option should be limited to cases where fine-tuning of the data path on guest applications is important.

Caution:

If the virtual CPUs cannot be allocated to run on the specified node on any compute node, the virtual machine may fail to launch

1. Declare the number of NUMA nodes to create on the guest.

   The following extra specification sets the number of NUMA nodes:

   ```
   hw:numa_nodes=n
   ```

   where *n* is the number of NUMA nodes to create (1 or more).

   For example, given the flavor **numa.pinned.asym**, use the following command to create two NUMA nodes.

   ```
   ~(keystone_admin)$ nova flavor-key numa.pinned.asym set hw:numa_nodes=2
   ```

   To set an image property instead, you can use the following command:

   ```
   ~(keystone_admin)$ nova image-meta image hw_numa_nodes=2
   ```

2. Optional: For each NUMA node, assign a list of CPUs.

   This step is optional. By default, available CPUs for the flavor are distributed evenly across the available NUMA nodes.

   The following extra specification assigns CPUs to a NUMA node:

   ```
   hw:numa_cpus.vnode_id=cpu_list
   ```

   where

   vnode_id

   is the number used to identify the virtual NUMA node (0 for the first node, 1 for the second, and so on).

   cpu_list

   is a comma-separated list of vCPUs to assign to the node. The vCPUs for the flavor are enumerated beginning with 0.

   For example, given flavor **numa.pinned.asym**, use the following command to assign vCPU 0 to the first NUMA node, and vCPUs 1 and 2 to the second NUMA node:

   ```
   ~(keystone_admin)$ nova flavor-key numa.pinned.asym set hw:numa_cpus.0=0 \
   hw:numa_cpus.1=1,2
   ```

3. Optional: For each NUMA node, assign an amount of memory.

   This step is optional. By default, available memory for the flavor is distributed evenly across the available NUMA nodes.

   The following extra specification assigns memory to a NUMA node:

   ```
   hw:numa_mem.vnode_id=ram_size
   ```

   where

   vnode_id

   is the number used to identify the virtual NUMA node (0 for the first node, 1 for the second, and so on).

   ram_size

   is the amount of RAM in MB.

   For example, given flavor **numa.pinned.asym**, use the following command to assign 512 MB of RAM to each of two NUMA nodes:

   ```
   ~(keystone_admin)$ nova flavor-key numa.pinned.asym set hw:numa_mem.0=512
    \
   hw:numa_mem.1=512
   ```

4. Optional: Specify whether memory for the flavor can be drawn from more than one NUMA node if necessary.

   This step is optional. By default, memory is allocated from the designated host NUMA node only.

   To control whether memory for the flavor can be drawn from more than one host NUMA node, use the following extra specification:

   ```
   hw:numa_mempolicy=policy
   ```

   where policy is either **strict** (to use only memory from the designated host NUMA node) or **preferred** (to permit memory from other host NUMA nodes to be used if necessary).

To pin the virtual NUMA nodes to host NUMA nodes, see *Pinning a Guest NUMA Node to a Host NUMA Node* on page 151.

Viewing the NUMA Node Configuration for a VM

You can use the CLI to display the NUMA node configuration for a VM.

Use the following command:

```
~(keystone_admin)$ nova show instance
```

where **instance** is the name or UUID of the instance.

Pinning a Guest NUMA Node to a Host NUMA Node

You can use flavor extra specs or image properties to pin a guest NUMA node to a host NUMA node.

By default, when instances are launched or migrated, the virtual NUMA nodes defined for the VMs are mapped to available host NUMA nodes. You can optionally designate a specific host NUMA node for a virtual NUMA node, using either a flavor extra specification (which takes the general form **hw:specification**), or an image property (which takes the general form **hw_specification**). This enables you to co-locate VM processes with AVS vSwitch processes for high-performance networking.

For information about assigning vSwitch processes to host NUMA nodes, see *Processor* on page 118.

For a VM with only one virtual NUMA node, you can use an extra specification to specify the host NUMA node. To add this extra spec to a flavor using the web administration interface, use the **NUMA Node** selection in the **Create Flavor Extra Spec** drop-down menu. This provides fields to associate a **Guest NUMA Node** with a **Host NUMA Node**. To access this menu, see *Flavor Extra Specifications* on page 143.

You can also pin NUMA nodes using the CLI. The following extra specification assigns a specific host NUMA node to a virtual NUMA node:

```
hw:numa_node.vnode_id=pnode_id
```

where

vnode_id

is the number used to identify the virtual NUMA node (0 for the first node, 1 for the second, and so on).

pnode_id

is the number used to identify the host NUMA node (0 for the first node, 1 for the second, and so on).

For example, given flavor **numa.pinned.asym**, use the following command to assign virtual NUMA node 0 to host NUMA node 1:

```
~(keystone_admin)$ nova flavor-key numa.pinned.asym set hw:numa_node.0=1
```

Affining a VM to a NUMA Node with a vSwitch Core

You can select whether to place a VM in the same NUMA node as a vSwitch (AVS) core.

For optimal networking performance, you can set an extra spec so that a VM is instantiated on a host only if the host can offer a NUMA node with at least one core assigned as a vSwitch. You can optionally specify best-effort placement, so that if a suitable host is unavailable, the VM is still instantiated, but using a NUMA node without a vSwitch core.

To add this extra spec to a flavor using the web administration interface, use the **Vswitch NUMA Affinity** selection in the **Create Flavor Extra Spec** drop-down menu. To access this menu, see *Flavor Extra Specifications* on page 143.

You can also specify the extra spec from the CLI by setting the following parameter for a flavor:

```
hw:wrs:vswitch_numa_affinity=vswitch_affinity
```

where *vswitch_affinity* is one of the following:

strict

Requires a host NUMA node that has a vSwitch core. For a VM configured to use multiple NUMA nodes, each host NUMA node must have a vSwitch core. If these conditions cannot be satisfied, the instantiation fails.

prefer

Prefers a host NUMA node that has a vSwitch core. For a VM configured to use multiple NUMA nodes, prefers host NUMA nodes with a vSwitch core for as many virtual NUMA nodes as possible. If these conditions cannot be satisfied, another NUMA node can be used.

For information about assigning vSwitch cores on a host, see *Processor* on page 118.

Specifying Best Effort for PCI NUMA Node Affinity

You can select whether to require direct NUMA-node access to a PCI interface when instantiating a VM.

By default, when a VM is configured to use an SR-IOV or PCI-passthrough interface, the VM is affined to a NUMA node that has direct hardware access to the physical PCI interface. If none is available, the instantiation fails.

You can set an extra spec so that if the requirement cannot be satisifed, the VM is instantiated on another NUMA node with mediated access to the interface.

To add this extra spec to a flavor using the web administration interface, use the **PCI NUMA Affinity** selection in the **Create Flavor Extra Spec** drop-down menu. To access this menu, see *Flavor Extra Specifications* on page 143.

You can also specify the extra spec from the CLI by setting the following parameter for a flavor:

```
hw:wrs:pci_numa_affinity=pci_affinity
```

where *pci_affinity* is one of the following:

strict

(default) Requires a host NUMA node with direct access to a PCI interface. For a VM configured to use multiple NUMA nodes, at least one host NUMA node must have direct PCI access. If these conditions cannot be satisfied, the instantiation fails.

prefer

Prefers a host NUMA node with direct access to a PCI interface. If none is available, another NUMA node can be used.

For more information about using PCI interfaces, see *PCI Passthrough Ethernet Interfaces* on page 29 and *SR-IOV Ethernet Interfaces* on page 33.

Specifying a Page Size for a VM

You can request a specific memory page size for a VM by using a flavor with the required extra spec, or by defining an image with the required metadata property.

Memory requested by a guest is allocated as one or more pages of a specified size. Once allocated, the memory is unavailable for use by other guests until the instance is terminated.

To add this extra spec to a flavor using the web administration interface, use the **Memory Page Size** selection in the **Create Flavor Extra Spec** drop-down menu. To access this menu, see *Flavor Extra Specifications* on page 143.

You can also specify the extra spec from the CLI by setting the following parameter for a flavor:

```
hw:mem_page_size=pagesize
```

where *pagesize* is one of the following:

small

Requests the smallest available size on the compute node, which is always 4KiB of regular memory.

large

Requests the largest available huge page size, 1GiB or 2MiB.

any

Requests any available size, including small pages. HP Helion OpenStack Carrier Grade uses the largest available size, 1GiB, then 2MiB, and then 4KiB.

2048

Requests a huge page of size 2 MiB. This is the default value when the extra spec is not used.

1048576

requests a huge page of size 1GiB.

If any other value is supplied, it is ignored and the default value of **2048** is used.

It is suggested that NFV applications use explicit huge page sizes to ensure predictable memory access at run time. Use of **large** and **any** sizes may cause VM migration issues when the available page size on the destination node is different.

For example, to set a 1 GiB huge page size on a flavor that has already been created, use the following command:

```
~(keystone)admin)$ nova flavor-key flavor_name set hw:mem_page_size=1048576
```

You can also define an image with the required property by including the hw_mem_page_size parameter, as in the following example:

```
~(keystone)admin)$ nova image-meta image set hw_mem_page_size=pagesize
```

Note that if you use image metadata to request a page size, the image is unable to access a **large** page unless the setting for the flavor is **large** or **any**.

Host memory on compute nodes reserved for use by VMs is partitioned by default using only 2 MiB huge page blocks. These blocks are used by default as the backing mechanism for VM virtual memory allocation requests. See *Host Memory Provisioning* on page 122 for details on how to modify the memory partitioning scheme for a host.

Specifying Local Storage for VM Ephemeral Resources

To use local storage for ephemeral resources, you must specify instantiation on a host with local storage. You can do this by using a flavor with the appropriate extra specification.

Each new flavor is automatically assigned a **Storage Type** extra spec that specifies, as the default, instantiation on hosts without local storage. You can change this extra spec so that hosts with local storage are used. Ephemeral storage resources (**Ephemeral Disk** and **Swap Disk** space) defined for the flavor use this local storage, which is allocated from the Local Volume Group on the host.

If the extra spec is set for local storage and instance is booted from an image, then the root disk is ephemeral and also uses local storage. If the instance is booted from a volume, the root disk uses Cinder-based storage allocated from the controller (for a system using LVM) or from storage hosts (for a system using 3PAR).

Note:

For faster VM booting, use a prepared Cinder volume. For more information, see *Creating Cinder Volumes for Boot Images* on page 168.

Before you can use local storage for a VM, you must configure at least one compute node to provide local storage. For more information, see *Configuring a Compute Host to Provide Local Storage* on page 64.

Caution:

Local storage is ephemeral.

- Unlike Cinder-based storage, local storage does not persist if the instance is terminated or the compute node fails.
- Live migration is not currently supported for an instance using local storage. Locking the host initiates a cold migration. The local storage for the instance is rebuilt.
- Resizing of local storage is partly supported. If the reconfigured instance is instantiated on the same host, then any root or ephemeral disks that use local storage are resized with their data preserved. Swap disks that use local storage are rebuilt. If the instance is migrated to another host, only cold migration is supported. All local storage for the instance is rebuilt.

To change the extra specification using the web administration interface, click **Edit** for the existing **Storage Type** extra specification, and then select the **Local Storage** check box. To access the extra specification, see *Flavor Extra Specifications* on page 143.

Caution:

If the **Storage Type** extra spec is not specified, instantiated VMs may use hosts with or without local storage. To ensure that instantiated VMs use storage of a known type, do not delete this extra spec.

You can also specify the extra spec from the CLI by setting the following parameter for a flavor:

```
aggregate_instance_extra_specs:localstorage=loc_storage
```

where *loc_storage* is one of the following:

true

Specifies hosts with local storage for use by the VM.

false

Specifies hosts without local storage for use by the VM.

For example:

```
~(keystone_admin)$ nova flavor-key flavor_name \
set aggregate_instance_extra_specs:localstorage=false
```

The local storage key is added by default on flavor creation and set to false (hosts with no local storage).

Enabling the Guest Heartbeat API for a VM

You can accommodate the use of guest heartbeats on a VM using an extra specification.

Select this option when you expect one or more of the guest applications running on the virtual machine to make use of the HP Helion OpenStack Carrier Grade Guest Heartbeat API. If no extra spec is added, then by default the API is disabled.

If this option is selected, the controller node starts heartbeat application-level polling cycles on virtual machines launched using the flavor. For more information about the Guest Heartbeat API, refer to the *HP Helion OpenStack Carrier Grade Software Development Kit*.

A guest application modified to use the Guest Server Heartbeat API can be more accurately monitored by internal messaging within the virtual machine. For more about application monitoring, see *Virtual Machines and Carrier-Grade Availability* on page 166.

To add this extra spec to a flavor using the web administration interface, use the **Guest Heartbeat** selection in the **Create Flavor Extra Spec** drop-down menu. To access this menu, see *Flavor Extra Specifications* on page 143.

Note:

The non-default value **True** is offered when the extra spec is added using the web administration interface.

To enable the Guest Heartbeat API using the CLI, use the following command:

```
~(keystone_admin)$ nova flavor-key flavor_name set
 sw:wrs:guest:heartbeat=value
```

where *value* is either **True** or **False**. If any other value is supplied, it is ignored and the default value **False** is used.

Enabling Server Group Messaging for a VM

You can enable a messaging API for a VM, for use with other VMs in the same server group.

If this extra spec is not added, then by default this API is disabled.

Server Group Messaging is a service that provides simple low-bandwidth datagram messaging and notifications for virtual machines that belong to the same server group. This message channel is available regardless of whether IP networking is functional within the server, and requires no knowledge about the other members in the group.

Guest applications can access this service if the Server Group Messaging API is enabled for the VM. To enable the API, select this option. For more information about the Server Group Messaging API, refer to the *HP Helion OpenStack Carrier Grade Software Development Kit*.

Note:

The non-default value **True** is offered when the extra spec is added.

The service provides three types of messaging:

- **Broadcast**—allows the server to send a datagram of up to 3050 bytes to all other servers in the server group.
- **Notification**—provides servers with information about changes to the state of other servers in the server group.
- **Status**—allows the server to query the current state of all servers in the group (including the originating server).

Caution:

This service is not intended for high-bandwidth or low-latency operations. If reliability is an important consideration, use acknowledgements and retries.

To add this extra spec to a flavor using the web administration interface, use the **Server Group Messaging** selection in the **Create Flavor Extra Spec** drop-down menu. To access this menu, see *Flavor Extra Specifications* on page 143.

To enable the Server Group Messaging API using the CLI, use the following command:

```
~(keystone_admin)$ nova flavor-key flavor_name set
 sw:wrs:srv_grp_messaging=value
```

where *value* is either False or True.

Disabling Instance Auto Recovery

You can disable auto recovery of failed virtual machine instances.

If this extra spec is not added, then the system will automatically recover failed virtual machine instances. Set this option to false if, for example, you are writing a Virtual Network Function Manager (VNFM) and the VNFM is to manage the auto-recovery of its VMs. Setting it to false disables this behavior. To re-enable virtual machine instance auto recovery, set the value to true.

Instance auto-recovery can be configured as an extra spec on a flavor, such that any VM instance created with this flavor will have the specified instance auto-recovery. Alternatively, instance auto-recovery can be configured as meta-data on an image, such that any VM instance created based on the image will have the specified instance auto-recovery. Note that Flavor extra-specs will override the image meta-data.

To set the option using the web administration interface, do one of the following:

- As *admin*, use the **Instance Auto Recovery** selection in the **Create Flavor Extra Spec** drop-down menu. To access this menu, see *Flavor Extra Specifications* on page 143.

- As either *admin* or a tenant, select **System** > **Images** > **Edit Image** > for the desired images and uncheck the **Instance Auto Recovery** option.

  To set the option from the command line do one of the following.

  - As *admin*, add an extra spec to a flavor:

    ```
    ~(keystone)admin)$ nova flavor-key flavorName set
     sw:wrs:auto_recovery=false
    ```

  - As *admin* or a tenant:

    ```
    ~(keystone)admin)$ glance image-update --property
     sw_wrs_auto_recovery=false imageName
    ```

Specifying a Live Migration Timeout

Live migrations can take a long time to complete for reasons such as high levels of guest activity, network latency, and so on. You can specify the number of seconds to wait for a live migration to complete.

If no value is set, the default is 800 seconds.

If the value is exceeded, the migration is canceled and a customer log entry is created. The counter starts when the migration begins. The minimum timeout value is 120 seconds and the maximum is 800 seconds. Set this value to 0 to disable the live migration timeout feature. For more information, see *Live Migration of Virtual Machines* on page 161.

Note:

NOVA maintains an internal timer which guards a complete lack of VM instance live migration progress for longer than 150 seconds. If this timer expires, the live migration will be canceled; regardless of the value of the live migration maximum timeout.

A live migration timeout can be configured as an extra spec on a flavor, such that any VM instance created with the flavor will inherit the specified timeout value. Alternatively, a live migration timeout can be configured as meta-data on an image, such that any VM instance created based on this Image will have the specified timeout value. If the timeout value is provisioned in both the flavor and the image, the smaller value is used.

To set this option using the web administration interface:

- As *admin*, use the **Live Migration Timeout** selection in the **Create Flavor Extra Spec** drop-down menu. To access this menu, see *Flavor Extra Specifications* on page 143.

or

- As either *admin* or a tenant, select **System** > **Images** > **Edit Image** > **Update Metadata** for the desired images and add an entry with:

  ```
  hw_wrs_live_migration_timeout=seconds
  ```

  Set the timer from the command line:

  The following examples set the timeout to 400 seconds.

  - As *admin*, add an extra spec to a flavor by:

    ```
    ~(keystone)admin)$ nova flavor-key flavorName set
     hw:wrs:live_migration_timeout=400
    ```

  - As *admin* or a tenant, update the metadata of an image by:

    ```
    ~(keystone)admin)$ glance image-update --property
     hw_wrs_live_migration_timeout=400 imageName
    ```

Specifying Live Migration Maximum Downtime

You can change the maximum amount of downtime to tolerate during a live migration.

If this extra spec is not added, then the default 500 milliseconds is used. The minimum timer value is 100 milliseconds.

Note:

Downtime during a live migration is the period during which the VM will be suspended in order to journal the final increment of data between source VM instance and destination VM instance, resulting in a temporary service interruption. Adjust this timer to avoid excessive down times that may violate service level agreements. When this threshold cannot be achieved within the *live_migration_timeout*, the migration is aborted. For more information about *live_migration_timeout*, see *Specifying a Live Migration Timeout* on page 156.

The live migration process will actually attempt to achieve a much smaller downtime than specified by this parameter by starting with a lower value approximately 1/10th of specified value, and increasing the downtime target and every 75 seconds for a total of 10 attempts. Setting the Live Migration Maximum Timeout, as described in *Specifying a Live Migration Timeout* on page 156, lower than 750 seconds results in the actual maximum tolerated live migration downtime being shorter than specified by this parameter.

Live migration maximum downtime can be configured as an extra spec on a flavor, such that any VM Instance created with the flavor will inherit the specified value. Alternatively, it can be configured as meta-data on an image, such that any VM instance created based on this image will have the specified value. If live migration maximum downtime is set in both the flavor and the image, the value from the flavor overrides the value from the image.

To set this option using the web administration interface:

- As *admin*, use the **Live Migration Max Downtime** selection in the **Create Flavor Extra Spec** drop-down menu. To access this menu, see *Flavor Extra Specifications* on page 143.

or

- As either *admin* or a tenant, select **System** > **Images** > **Edit Image** > **Update Metadata** for the desired images and add an entry with:

  ```
  hw_wrs_live_migration_max_downtime=milliseconds
  ```

  Set the timer from the command line:

  The following examples set the timeout to 350 milliseconds.

  - As *admin*, add an extra spec to a flavor:

    ```
    ~(keystone)admin)$ nova flavor-key flavorName set
     hw:wrs:live_migration_max_downtime=350
    ```

  - As *admin* or a tenant, update the metadata of an image:

    ```
    ~(keystone)admin)$ glance image-update --property
     hw_wrs_live_migration_max_downtime=350 imageName
    ```

Isolating an Untrusted Guest

You can isolate the network resources used and accessible by a potentially untrusted virtual machine instance.

This ensures that the untrusted virtual machine cannot exhaust or corrupt shared network resources and impact the operation or performance of other virtual machines hosted on the same compute node. Note that network resource isolation comes at a slight degradation of network performance for this guest.

Set this option to true when you want to isolate the network resources used and accessible by an untrusted guest instance from all other instances running on the host. Set it to false to turn isolation off. If this extra spec is not set, then the instance's NICs use of network resources are not isolated.

To add this extra spec to a flavor using the web administration interface, use the **NIC Isolation** selection in the **Create Flavor Extra Spec** drop-down menu. To access this menu, see *Flavor Extra Specifications* on page 143. This enables isolation of a virtual machine instance's NICs' use of network resources from all other physical or virtual interfaces on the host. If this extra spec is not set, then the instance's network interfaces are not isolated.

To use this feature, the compute host must have sufficient 2 MB huge pages available for isolated NIC creation. For example, a guest with 3 NICs would require that 48 MB of 2 MB huge pages be available on its host. Isolated guests will only be launched on hosts that meet this requirement. For more information on using huge pages, see *Host Memory Provisioning* on page 122.

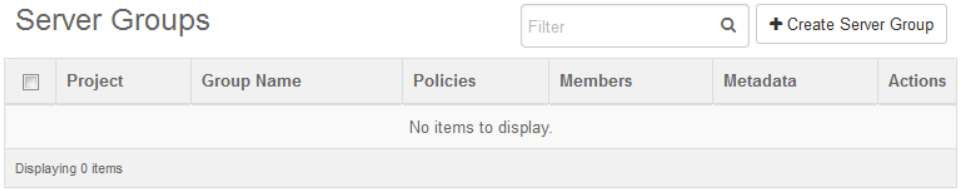
Set this option using the CLI:

```
~(keystone_admin)$ nova flavor-key flavorName set
 hw:wrs:nic_isolation=true
```

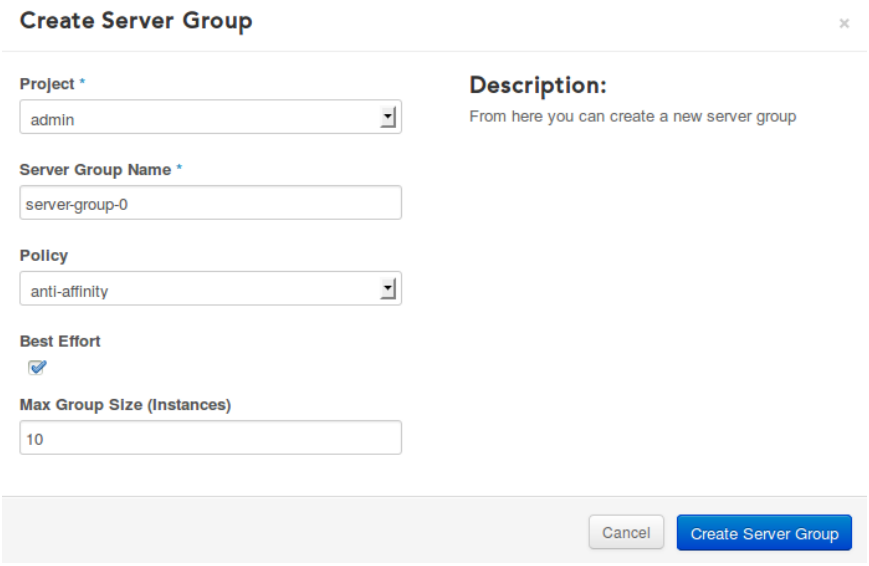
Server Groups

Server Groups is a mechanism to group virtual machines to which a common set of attributes is applied.

Select the option Server Groups from either the **Project** or the **Admin** tabs on the web administration interface to display the **Server Groups** window illustrated in the following figure:



Click the button **Create Server Group** to create a new server group, as illustrated below:



The following parameters can be defined for the new server group:

Project

Identifies the tenant the new service group should be associated with. This field is only available as part of the **Admin** operations; the project identity is implicit within the context of a specific tenant.

Server Group Name

The name for the new server group.

Policy

The following scheduling policy options are available:

affinity

When this option is selected, new instances launched as part of this server group are scheduled to run on the same compute node.

anti-affinity

When this option is selected, new instances launched as part of this server group are scheduled to run on different compute nodes. For example, you can use the anti-affinity option when you have two virtual machines that run in 1:1 HA protection mode, and you want them to be protected against hardware failures.

See *The Virtual Machine Scheduler* on page 160 and *Live Migration of Virtual Machines* on page 161 for additional considerations on scheduling and live migration for each of these policies.

Best Effort

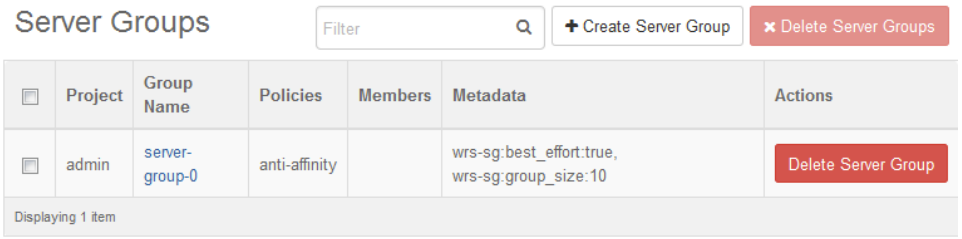
When selected, the policy in place, affinity, or anti-affinity, is enforced whenever possible, on a best-effort basis. If for any reason the policy cannot be enforced, then the deployment of the new instance proceeds using the regular scheduling.

When cleared, the policy in place must be executable for the new instance to be scheduled. Launching of a new virtual machine fails if there are no resources to comply with the selected policy.

Max Group Size (Instances)

Determines the maximum number of instances that can co-exist as part of this server group.

The new service group is displayed as follows:



Once a server group is defined, you can add virtual machines to it at launch time. This is done by selecting the desired server group from the tab **Server Group** on the launch window.

Server Groups and the CLI

Server groups can be created and deleted using the CLI as illustrated in the following examples:

```
~(keystone_admin)$ nova server-group-create --policy affinity \
--metadata wrs-sg:best_effort=true --metadata wrs-sg:group_size=2 ht-group-
test
+--------------+--------------+---------------+---------+------------+
| Id           | Name         | Policies      | Members | Metadata   |
+--------------+--------------+---------------+---------+------------+
| 63752c24-... | ht-group-test | [u'affinity... | []      | {u'best... |
+--------------+--------------+---------------+---------+------------+
~(keystone_admin)$ nova server-group-delete 63752c24-...
Server group 63752c24-... has been successfully deleted.
```

The Virtual Machine Scheduler

The virtual machine scheduler in HP Helion OpenStack Carrier Grade is a modified version of the OpenStack Nova scheduler. It allows for better placement of virtual machines in order to exploit hardware and network resources, and for a better distribution of the system workload overall.

In addition to the scheduler actions taken based upon the virtual machine flavor in use, as discussed in *Virtual Machine Flavors* on page 142, the following are enhancements integrated into the HP Helion OpenStack Carrier Grade Nova scheduler:

Memory over-commit policy

There is no support for memory over-commit when scheduling virtual machines. When launching a virtual machine, the Nova scheduler reserves the requested memory block from the system in its entirety. The launch operation fails if not enough memory can be allocated from the compute cluster.

Network load balancing across processor nodes

When scheduling threads for new virtual CPUs, the Nova scheduler selects the target core taking into account the load average of the running AVS cores on each processor node. In a common scenario, the AVS is deployed on two cores, not necessarily part of the same processor. When a new virtual machine is launched, the Nova scheduler looks at the average load incurred by AVS cores on each processor, and then selects the processor with the lighter load as the target for the new virtual CPUs.

This scheduling approach aims at balancing the switching load on the AVS across virtual machines as they are deployed, and minimizing the cross-NUMA inefficiencies for AVS switching.

Provider networks access verification

When scheduling a virtual machine, the Nova scheduler verifies that the target compute node has data interfaces on all needed provider networks, as determined by the list of tenant networks the virtual NICs are attached to. This verification helps prevent unnecessary debugging steps incurred when the networking services on a guest application fail to operate due to the lack of proper network access.

NUMA node pinning

When scheduling a virtual machine, the Nova scheduler selects a host on which the virtual NUMA nodes can be affined to specific host NUMA nodes according to the flavor extra specifications or image properties.

The Nova Scheduler and Server Group Policies

Virtual machines launched as part of a Server Group are subject to scheduling actions determined by the selection of scheduling policy. See *Server Groups* on page 158 for details.

Affinity policy

The goal of this policy is to schedule all virtual machines in the Server Group to execute on the same host. The target compute node is selected by the Nova scheduler when the first instance in the Server Group is launched, in compliance with its corresponding flavor, network, and system requirements.

If the **Best Effort** flag of the Server Group is clear, then the scheduling policy is strictly enforced. Any new instance in the Server Group will fail to launch if any run-time requirements cannot be satisfied by the selected compute node.

If the **Best Effort** flag of the Server Group is set, then the scheduling policy is relaxed. New instances are scheduled to run on the selected compute node whenever possible, but are scheduled on a different host if otherwise necessary.

Anti-affinity policy

The goal of this policy is to schedule each virtual machine in the Server Group to execute on a different host. The target compute node for each instance is selected by the Nova scheduler in compliance with the corresponding

flavor, network, and system requirements. As with the affinity policy, the Nova scheduler takes no special considerations regarding the nature of the target host, HT-enabled or not.

If the **Best Effort** flag of the Server Group is clear, then the scheduling policy is strictly enforced. Any new instance in the Server Group will fail to launch if its run-time requirements can not be satisfied by any newly selected compute node.

If the **Best Effort** flag of the Server Group is set, then the scheduling policy is relaxed. New instances are scheduled to run on a different compute node whenever possible, but are scheduled on any already selected host if otherwise necessary.

Live Migration of Virtual Machines

Live migration occurs when a virtual machine is transferred to execute on a different compute node with minimal disruption of the guest applications. This can happen automatically, or upon request by the system administrator.

Note:

Live migration is not currently supported for instances using the following:

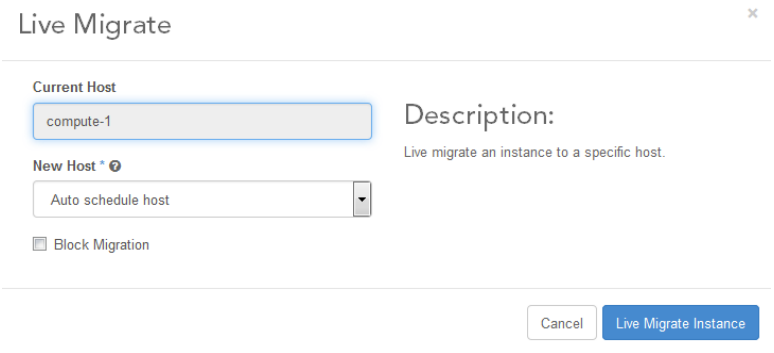
- PCI passthrough
- SR-IOV
- Local storage

While executing a live migration operation, HP Helion OpenStack Carrier Grade manages the virtual machine's state in such a way that it appears unmodified on the migrated instance. This includes:

- system memory, both kernel and user space
- access to all non-local storage resources, including LVM/iSCSI and Cinder
- all the virtual machine networking options (unmodified/virtio, AVP kernel driver, AVP DPDK Poll Mode Driver), and AVS

Automatic migration of virtual machines occurs whenever the administrator initiates a locking operation on a compute node. In this case, HP Helion OpenStack Carrier Grade first live-migrates all virtual machine instances off of the compute node before administratively locking it.

The **admin** user can also initiate live migrations manually from the **Instances** page available by clicking the option Instances on the **Admin** side pane of the web management interface. The More button of the selected instance provides the option Live Migrate Instance. When selected, the **Live Migrate** window is displayed as illustrated below:



The following fields are available:

Current Host

A read-only field displaying the compute node the selected instance is currently running on.

New Host

The target compute node for the migration. The default value is to let HP Helion OpenStack Carrier Grade auto-schedule the virtual machine following the current scheduling guidelines and constraints. Optionally, you can manually select the target compute node.

Note that the set of available target compute nodes for the migration is still subject to the scheduler constraints from the virtual machine flavor and other systems options that might be in place.

Live Migration and Server Group Policies

Virtual machines launched as part of a Server Group are subject to additional live migration restrictions determined by the selection of scheduling policy. See *Server Groups* on page 158 for details.

Affinity policy

The goal of this policy is to schedule all virtual machines in the Server Group to execute on the same host.

If the **Best Effort** flag of the Server Group is clear, then the individual instances cannot be migrated since this would break the affinity policy.

Note that this means that a compute node running instances in a Server Group with affinity policy in strict mode cannot be locked. An alternative mode of operation is to always set the **Best Effort** flag and then manually migrate the instances to a common host.

If the **Best Effort** flag of the Server Group is set, then any individual instances can migrate to any other available host.

Anti-affinity policy

The goal of this policy is to schedule each virtual machine in the Server Group to execute on a different host.

If the **Best Effort** flag of the Server Group is clear, then the individual instances can migrate provided that there are suitable hosts where no other Server Group instance is running.

If the **Best Effort** flag of the Server Group is set, then any individual instances can migrate to any other available host.

Scaling Virtual Machine Resources

You can scale the resources of individual instances up or down.

Currently, the CPU resources for an instance are scalable.

For an instance to be scalable, the following requirements must be satisfied:

- The image used to launch the instance must support scaling.

  The example image provided with HP Helion OpenStack Carrier Grade supports scaling. You can also build your own image, incorporating the required libraries and services. For more about building your own images, or about the technical requirements for scaling support, refer to the documentation included with the HP Helion OpenStack Carrier Grade Software Development Kit.
- The flavor used to launch the instance must be configured with maximum and minimum scaling limits (the *scaling range*) for the resource.

When scaling a VM, use the credentials of the user that launched the VM. This ensures that quotas are correctly managed.

Depending on the resource being scaled, the scaling behavior applied by HP Helion OpenStack Carrier Grade may be adjustable. For example, you can control which CPUs are released when the CPU resources for an instance are scaled down. To adjust scaling behavior, use the App Scale Helper script.

Normally, scaling is performed under the direction of Heat orchestration. For more about Heat autoscaling, see *Resource Scaling (Autoscaling)* on page 226. If required, you can scale resources manually from the command line using the `nova scale` command, with the following syntax:

```
~(keystone_admin)$ nova scale instance_id resource {up | down}
```

For example, to reduce the number of CPUs allotted to an instance, you can use this command:

```
~(keystone_admin)$ nova scale instance_id cpu down
```

Note:

To scale up the resources for an instance, sufficient resources are required on the host. If all available resources are already allocated to other instances, you may need to free up resources manually.

The App Scale Helper Script

The App Scale Helper script supports resource scaling customizations.

This is an optional script running on the guest. If present, it can modify aspects of scaling behavior. For example, it can control which CPU is taken offline during a scale-down operation, overriding the default selection.

It can also call other scripts or programs. You can use this to coordinate or manage processing work for the vCPUs as you scale them.

For more about the App Scale Helper script, refer to the documentation for the HP Helion OpenStack Carrier Grade Software Development Kit.

CPU Scaling

HP Helion OpenStack Carrier Grade supports CPU up/down scaling for instances.

You can enable CPU scaling for an instance by setting a scaling range (see *Setting the CPU Scaling Range* on page 163). You must also specify the use of dedicated CPUs (see *Specifying Dedicated CPUs for a VM* on page 147).

When an instance is first started, all available vCPUs are brought online. If the instance is restarted, the number of online vCPUs is set to match the number when the instance was stopped. This is controlled by a helper script that automatically takes the required number of vCPUs offline.

When CPU resources are scaled up, a vCPU is brought online from the pool of available vCPUs for the instance, subject to the maximum allowed by the flavor. The lowest-numbered offline vCPU is selected.

When CPU resources are scaled down, a vCPU is taken offline, subject to the minimum allowed by the flavor. By default, the highest-numbered online vCPU is selected. This can be overridden using the App Scale Helper script.

For more about CPU scaling, refer to the documentation for the HP Helion OpenStack Carrier Grade Software Development Kit.

Note:

If there are insufficient resources to launch a scalable VM with the expected allotment when the VM is started or restarted, the launch fails. The VM is *not* automatically scaled down to compensate.

Setting the CPU Scaling Range

You can define the maximum and minimum CPU resources available to an instance by using a flavor.

- To set the maximum number of VCPUs for an instance, define a flavor with the required number of vCPUs.
- To set the minimum number of vCPUs, edit the flavor to include an Extra Spec. The minimum cannot be less than one.

You can use the web administration interface or the CLI to edit the flavor. The CLI parameter for setting the minimum number of CPUs is as follows:

```
hw:wrs:min_vcpus=min
```

where **min** is the minimum number of CPUs.

For example:

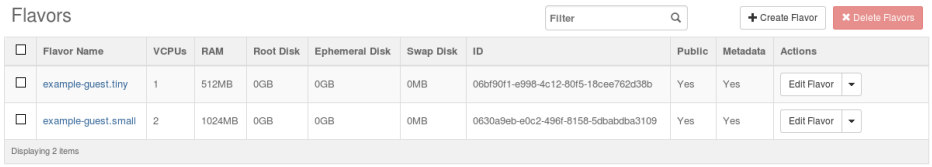
```
~(keystone_admin)$ nova flavor-key flavor_name set
 hw:wrs:min_vcpus=integer_value
```

For complete information about working with flavors, see *Virtual Machine Flavors* on page 142.

For CPU scaling to take effect, the **CPU Policy** for the VM must be set to **Dedicated**. For more information, see *Specifying Dedicated CPUs for a VM* on page 147.

1. Display the **Flavors** list.

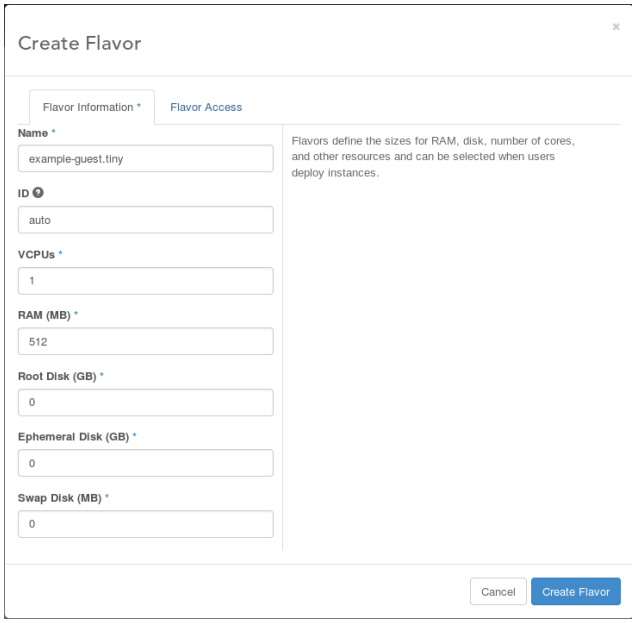
   Select the **Admin** menu in the web administration interface, and then in the **System Panel** section of the menu, click **Flavors**.



2. Optional: If necessary, create a new flavor.

   If a suitable flavor already exists, you can edit it to specify the maximum and minimum vCPUs.

   If no suitable flavor exists, you can create a new flavor by clicking **Create Flavor** in the list to open the **Create Flavor** dialog box.



   Assign a **Name** for the flavor, and then save the flavor by clicking **Create Flavor** in the dialog box.

3. Specify the maximum number of vCPUs for the instance.

   In the **Flavors** list, locate the flavor, and then click **Edit** to open the **Edit Flavor** dialog box.

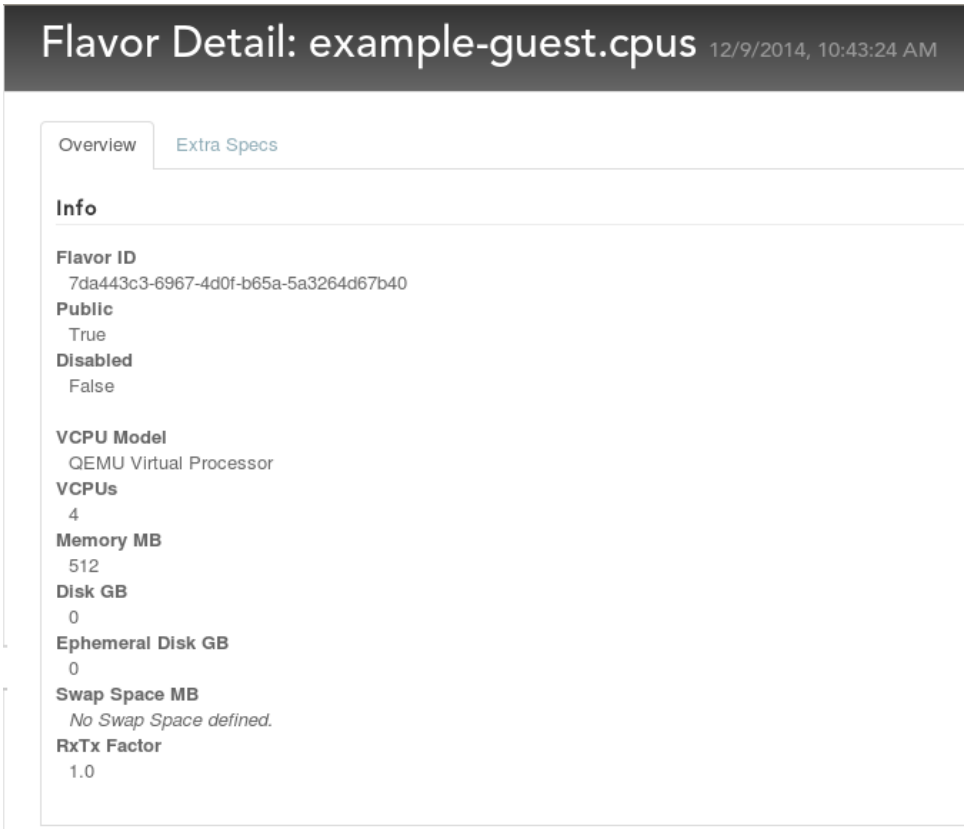
   In the **vCPU** field, enter the maximum number of vCPUs.

   When you are finished editing, click **Save** to return to the **Flavors** list.

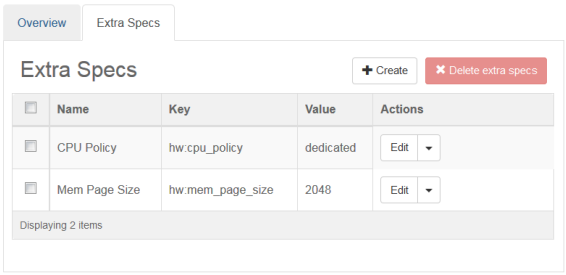
4. Specify the minimum number of vCPUs for the instance.

   You can specify the minimum number of vCPUs by adding an Extra Spec for the flavor.

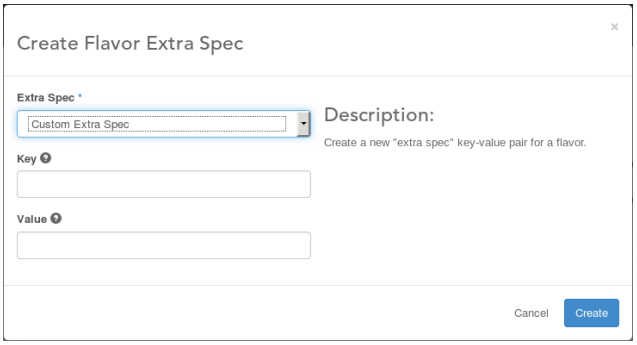
   a) In the **Flavors** list, click the **Flavor Name** for the flavor to open the **Flavor Details** dialog box.



b) On the **Extra Specs** tab, click **Create**.



c) In the **Create Flavor Extra Spec** dialog, select **Minimum Number of CPUs** from the **Extra Spec** drop-down menu.



d) In the **Key** field, enter wrs:min_vcpus.

e) In the **Value** field, enter the minimum allowed number of vCPUs for the flavor.

f) Click **Create**.

Virtual Machines and Carrier-Grade Availability

The HP Helion OpenStack Carrier Grade virtualized environment provides a health monitoring mechanism that can be used to implement and support the deployment of guest applications in Carrier-Grade High Availability (HA) mode.

A simplified view of the health monitoring system is illustrated in the following figure:

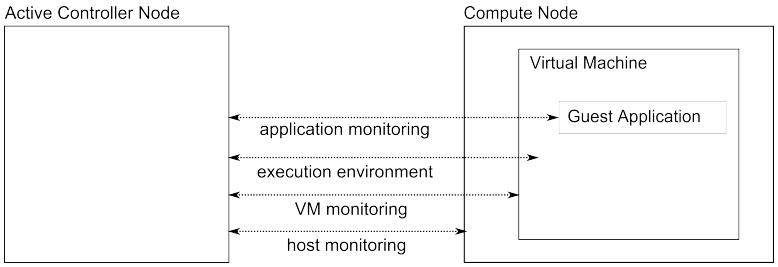


Figure 7: Hardware and software health monitoring

Host monitoring

A host failure occurs when the compute node hardware that hosts the application fails, or when its kernel becomes unresponsive. Host failures are detected by the active controller node through an efficient and scalable monitoring process that runs continuously during normal operation of the HP Helion OpenStack Carrier Grade cluster.

When such a failure is detected, HP Helion OpenStack Carrier Grade automatically re-schedules all affected virtual machines for deployment on alternative compute nodes.

Host monitoring, and its recovery mechanisms, are always available for every deployed virtual machine in the HP Helion OpenStack Carrier Grade Cluster.

Virtual machine monitoring

From the point of view of a guest application, a hardware failure occurs when the hosting virtual machine fails to execute. From the compute node, this means that the virtual machine process itself is experiencing execution problems, or that it is no longer running.

When such a failure is detected, HP Helion OpenStack Carrier Grade automatically tries to restart the affected virtual machine on the same compute node. If the restart operation fails, the virtual machine is automatically scheduled to deploy on an alternative compute node.

Virtual machine monitoring, and its recovery mechanisms, are always available for every deployed virtual machine in the HP Helion OpenStack Carrier Grade cluster.

Execution environment

The HP Helion OpenStack Carrier Grade Guest-Client daemon, built into the guest image, verifies the execution environment. If the guest kernel becomes unresponsive, the lack of heartbeat messages from the daemon triggers an alarm on the active controller.

For all purposes, the virtual machine is considered then to be at fault, and is therefore re-scheduled for execution, first on the same compute node, and then on a different host if necessary.

Application monitoring

Software failures at the application level can be addressed within the context of the following failure scenarios:

Sudden death of the application process

The application process ends abruptly, likely because of a software bug in its code.

The application becomes unresponsive

Several conditions can lead the application process to stall, that is, to be unscheduled for additional work. This may happen because the application is waiting for some resource that is not available, or because of a bug in the application's logic.

The application declares itself to have failed

Logic built into the application determines conditions under which it must declare itself to be in an unrecoverable error state.

Handling of these cases is optional, and subject to the proper API integration into the guest application itself.

Application monitoring happens when the HP Helion OpenStack Carrier Grade Guest-Client daemon in the virtual machine registers the application for monitoring. What happens after an application failure occurs is determined when the application is registered. By default, an application failure triggers a hard reboot of the virtual machine instance.

If the application makes use of the HP Helion OpenStack Carrier Grade Guest Heartbeat API, it can instruct the active controller on how to proceed when the application declares itself in a state of error. This could include a restart of the virtual machine, which in the case of 1:1 HA application/VM pair, would trigger a VM switchover. The application can also use the *VM Peer Notification API* to receive notifications when virtual machines in the same server group go up or down.

Refer to the *HP Helion OpenStack Carrier Grade SDK* for more information on how to configure the Guest-Client daemon and the use of the HP Helion OpenStack Carrier Grade Guest Heartbeat API.

Data persistence is another aspect that impacts the high-availability of a running application. This is addressed by providing storage persistence across applications restarts for Cinder remote HA block storage using distributed storage backends, such as 3PAR, for configuration databases.

By instrumenting the level of interaction between the guest application and the health-monitoring system, and by using the appropriate Cinder storage backends, guest applications can be deployed to run in different HA scenarios. Note that in all cases, monitoring of hardware and virtual machines, and their corresponding failure recovery mechanisms, are always active.

HA-unaware Guest Applications

These are applications designed with no consideration for special behavior needed when a failure occurs. Typical scenarios include:

- Stateless applications such as web servers serving static data, and data query systems. The requirement in these cases is for the application to be running continuously.

  This requirement is fulfilled when the application is deployed on HP Helion OpenStack Carrier Grade by the automatic monitoring at the hardware and virtual machine levels. Optionally, the application can benefit from all other monitoring levels when the proper instrumentation is in place.
- Stateful applications such as an enterprise-level call center, where the application's state is maintained in a database or some form of journaling system. An additional requirement in these cases is for storage persistence across restart operations, that is, for the state to be recoverable when the application restarts.

  This additional requirement is fulfilled when the application is deployed on HP Helion OpenStack Carrier Grade by the Cinder services, when configured to use distributed storage backends.

HA-aware Guest Applications

Typically, these are legacy applications supporting their own HA framework. They are expected to run unmodified, or with minimal changes, when deployed on HP Helion OpenStack Carrier Grade.

As with any other application, HA-aware applications benefit from the hardware and virtual machine monitoring processes, which provide them with automatic restart upon failures. When coupled with server groups (see *Server Groups* on page 158), they also benefit from the anti-affinity options to ensure that hot/standby running applications are protected against hardware failures.

HA-aware applications also benefit from internal tenant networks on top of which the application's HA framework, and any journaling framework it may use, can be deployed.

Connecting Virtual Machines to External Networks

You can provide VMs with external connectivity using one of several methods.

You can use source network address translation (SNAT) or floating IP addresses to expose VMs to an external network. Alternatively, you can use an external NAT device, or use a network address space for the VMs that is directly visible from the external network.

You can associate a floating IP address with a VM using a pool of available floating IP addresses configured by the system administrator. When a host on an external network initiates a connection to a public floating IP address, the virtual router updates a connection table to enable bidirectional traffic flow between the host and the VM.

When SNAT is enabled, the virtual router updates the connection table only when a VM initiates connection to an external host. Until this happens, the VM is inaccessible from the external network. This is true even if the private IP address of the VM is used directly by incoming traffic.

You can use floating IP addresses and SNAT in any combination. The following table presents an overview.

	Floating IP disabled	**Floating IP enabled**
SNAT disabled	VMs are directly accessible from the external network, provided the tenant network address space is directly visible from the external network, or NAT is implemented using a separate device.	VMs cannot initiate connections to the external network unless they have assigned floating IP addresses. External hosts can initiate connections to VMs that have assigned floating IP addresses.
SNAT enabled	VMs can initiate connections to the external network. External hosts cannot initiate connections to the VMs.	VMs can initiate connections to the external network. External hosts can initiate connections to VMs that have assigned floating IP addresses.

To enable or disable SNAT, see *Configuring SNAT on a Virtual Router* on page 54.

To configure floating IP addresses for VMs, refer to the public OpenStack documentation.

To connect VMs to external networks, you require an internal tenant network for the VMs, an external tenant network connected to an edge router, and a virtual router attached to both networks. For more information about adding and configuring these components, see the *HP Helion OpenStack Carrier Grade Reference Deployment Scenarios*.

Creating Cinder Volumes for Boot Images

You can create Cinder volumes to store VM boot images.

The use of Cinder volumes is recommended for fast instance booting.

1. Log in as the appropriate tenant.
2. On the **Project** menu, open the **Compute** section, and then click **Volumes**.
3. On the **Volumes** page, click **Create Volume**
4. Complete the **Create Volume** form.

   For an example, see the *HP Helion OpenStack Carrier Grade Reference Deployment Scenarios: Creating a Volume*.

Launching Virtual Machine Instances

You can launch a virtual machine from the web administration interface or the CLI.

You can launch a virtual machine from the CLI using the `nova boot` command. For an example, see the *HP Helion OpenStack Carrier Grade Reference Deployment Scenarios: Launching Instances Using the CLI*.

To launch an instance from the web administration interface, use the following steps.

To support fast booting, a Cinder volume is recommended. To prepare one, see *Creating Cinder Volumes for Boot Images* on page 168.

1. Log in as the appropriate tenant.
2. On the **Project** menu, open the **Compute** section, and then click **Instances**.
3. On the **Instances** page, click **Launch Instance**.
4. Complete the **Launch Instances** form.

   For examples, see the *HP Helion OpenStack Carrier Grade Reference Deployment Scenarios: Deploying the Bridging Scenario* and *Deploying the Routing Scenario*.

   If the system uses compute hosts with local storage, ensure that the instances logical volume space on the hosts is configured appropriately. For more information, see *Instances Logical Volume Considerations* on page 69.

   Note:

   To use boot-from-image, you must enable local storage for the flavor using an extra specification. For more information, see *Specifying Local Storage for VM Ephemeral Resources* on page 153.

   Note:

   For fast instance booting, use a prepared Cinder volume, and set the **Instance Boot Source** to **Boot from Volume**.

Chapter 6

Fault Management

Topics:

You can examine and filter the alarms and logs generated by HP Helion OpenStack Carrier Grade in order to monitor and respond to fault conditions.

Using the **Fault Management** page on the **Admin** menu, you can access the following system records:

- **Active Alarms**—Alarms that are currently set, and require user action to clear them.
- **Historical Alarms**—Alarms that have been raised in the past, including those that have been cleared.
- **Customer Logs**—Events that do not require user action, but may provide useful information for fault management.

For more about active and historical alarms, see *System Alarms* on page 171. For more about customer logs, see *Customer Logs* on page 187.

System Alarms

HP Helion OpenStack Carrier Grade can generate system alarms when operational conditions change on any of the hosts in the cluster.

This includes alarms based on hardware sensors monitored by board management modules, if present. For more information, see *Sensors* on page 134.

The supported interfaces to the alarms subsystem are the command line interface (see *System Alarms CLI Commands* on page 176), the web interface (click the **Fault Management** menu on the **Admin** tab), SNMP, and the system REST API.

About SNMP Support

Support for SNMP is implemented as follows:

- access is disabled by default, must be enabled manually from the command line interface
- available using the controller's node floating OAM IP address, over the standard UDP port 161
- supported version is SNMPv2c
- access is read-only for all SNMP communities
- all SNMP communities have access to the entire OID tree, there is not support for *VIEWS*
- supported SNMP operations are *GET*, *GETNEXT*, *GETBULK*, and *SNMPv2C-TRAP2*
- the SNMP *SET* operation is not supported

SNMPv2-MIB (RFC 3418)

Support for the basic standard MIB for SNMP entities is limited to the System and SNMP groups, as follows:

- System Group, **.iso.org.dod.internet.mgmt.mib-2.system**
- SNMP Group, **.iso.org.dod.internet.mgmt.mib-2.snmp**
- coldStart and warmStart Traps

The following system attributes are used in support of the SNMP implementation. They can be displayed using the `system show` command.

contact

A read-write system attribute used to populate the **sysContact** attribute of the SNMP System group. The contact value can be set with the following command:

```
~(keystone_admin)$ system modify contact="the-site-contact"
```

location

A read-write system attribute used to populate the **sysLocation** attribute of the SNMP System group. The location value can be set with the following command:

```
~(keystone_admin)$ system modify location="some-location"
```

name

A read-write system attribute used to populate the **sysName** attribute of the SNMP System group. The name value can be set with the following command:

```
~(keystone_admin)$ system modify name="the-system-name"
```

software_version

A read-only system attribute set automatically by the system. Its value is used to populate the **sysDescr** attribute of the SNMP System group.

The following SNMP attributes are used as follows:

sysObjectId

Set to **iso.org.dod.internet.private.enterprise.wrs.titanium** (1.3.6.1.4.1.1.2).

sysUpTime

Set to the up time of the active controller.

sysServices

Set to the nominal value of 72 to indicate that the host provides services at layers 1 to 7.

Enterprise MIBs

HP Helion OpenStack Carrier Grade supports the *Enterprise Registration* and *Alarm* MIBs.

Enterprise Registration MIB, wrsEnterpriseReg.mib

Defines the Systems (WRS) hierarchy underneath the **iso(1).org(3).dod(6).internet(1).private(4).enterprise(1)**. This hierarchy is administered as follows:

- **.wrs(731)**, the IANA-registered enterprise code for HP Helion OpenStack Carrier Grade Systems
- **.wrs(731).wrsCommon(1).wrs<Module>(1-...)**, defined in `wrsCommon<Module>.mib`.
- **.wrs(731).wrsProduct(2-...)**, defined in `wrs<Product>.mib`.

Alarm MIB, wrsAlarmMib.mib

Defines the common TRAP and ALARM MIBs for HP Helion OpenStack Carrier Grade Systems products. The definition includes *textual conventions*, an *active alarm table*, a *historical alarm table*, and *traps*.

Textual Conventions

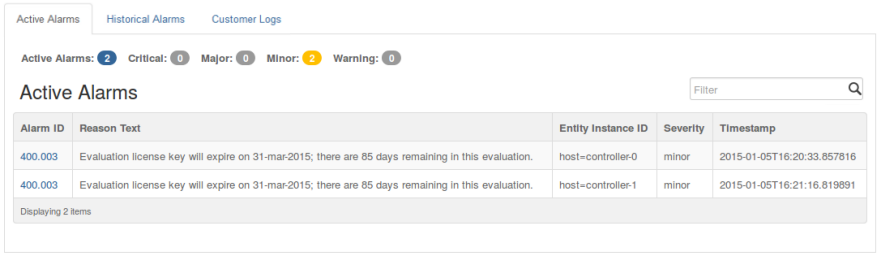
Semantic statements used to simplify definitions in the active alarm table and traps components of the MIB.

Active Alarm Table

A list of all *active* or *set* alarms in the system. Each entry in the table includes the following variables:

- *UUID*
- *AlarmID*
- *EntityInstanceID*
- *DateAndTime*
- *AlarmSeverity*
- *ReasonText*
- *EventType*
- *ProbableCause*
- *ProposedRepairAction*
- *ServiceAffecting*
- *SuppressionAllowed*

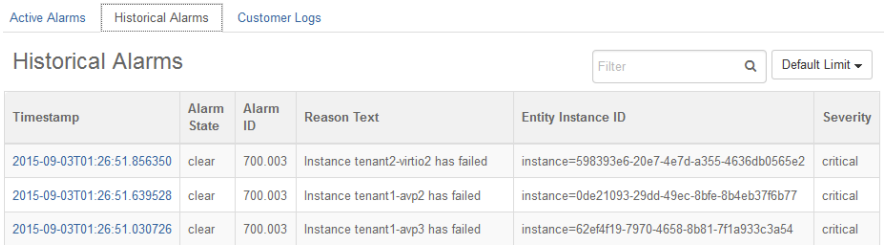
On the web interface, click the **Fault Management** menu on the **Admin** tab, and then select the **Active Alarms** tab. Active alarms are displayed as illustrated below:



Historical Alarm Table

A history of set and clear alarm operations in the system. The table includes the same variables as the active alarm table, plus the variable *AlarmState* used to indicate whether the table entry is a SET or a CLEAR operation.

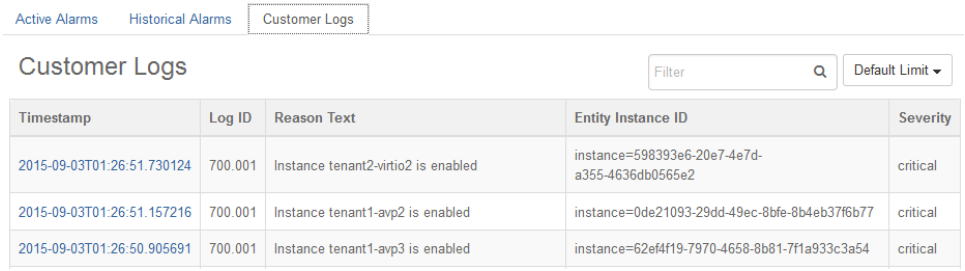
On the web interface, click the **Fault Management** menu on the **Admin** tab, and then select the **Historical Alarms** tab. The alarm history is displayed as illustrated below:



Customer Log Table

A list of all customer logs in the system. The table includes the same variables as the Active Alarm Table, with the exception that *ProposedRepairAction* and *SuppressionAllowed* are excluded.

On the web interface, click the **Fault Management** menu on the **Admin** tab, and then select the **Customer Logs** tab. Logs are displayed as illustrated below:



Traps

Defines the following generic traps:

- **wrsAlarmCritical**
- **wrsAlarmMajor**
- **wrsAlarmMinor**
- **wrsAlarmWarning**
- **wrsAlarmMessage**
- **wrsAlarmClear**
- **wrsAlarmHierarchicalClear**

For all alarms, the Notification Type is based on the severity of the trap or alarm. This is done to facilitate the interaction with most SNMP trap viewers which typically use the Notification Type to drive the coloring of traps, that is, red for critical, yellow for minor, and so on.

Customer Logs always result in **wrsAlarmMessage** traps.

For Critical, Major, Minor, Warning, and Message traps, all variables in the active alarm table are included as *varbinds*.

For the Clear trap, *varbinds* include only the *AlarmID*, *EntityInstanceID*, *DateAndTime*, and *ReasonText* variables.

For the HierarchicalClear trap, *varbinds* include only the *EntityInstanceID*, *DateAndTime*, and *ReasonText* variables.

Enabling SNMP Support

In order to have a workable SNMP configuration you must use the command line interface on the active controller to:

1. Define at least one SNMP community string. See *Adding an SNMP Community String* on page 174 for details.
2. Configure at least one SNMP trap destination so that alarms can be reported as they happen. See *Configuring SNMP Trap Destinations* on page 175 for details.

Adding an SNMP Community String

To enable SNMP services you need to define one or more SNMP community strings using the command line interface.

No default community strings are defined on HP Helion OpenStack Carrier Grade after the initial commissioning of the cluster. This means that no SNMP operations are enabled by default.

The following exercise illustrates the system commands available to manage and query SNMP community strings. It uses the string *commstr1* as an example.

All commands must be executed on the active controller's console, which can be accessed using the OAM floating IP address. You must acquire Keystone **admin** credentials in order to execute the commands.

1. Add the SNMP community string *commstr1* to the system.

```
~(keystone_admin)$ system snmp-comm-add -c commstr1
+-----------+--------------------------------------+
| Property  | Value                                |
+-----------+--------------------------------------+
| access    | ro                                   |
| uuid      | eccf5729-e400-4305-82e2-bdf344eb868d |
| community | commstr1                             |
| view      | .1                                   |
+-----------+--------------------------------------+
```

The following are attributes associated with the new community string:

access

The SNMP access type. In HP Helion OpenStack Carrier Grade all community strings provide read-only access.

uuid

The UUID associated with the community string.

community

The community string value.

view

The is always the full MIB tree.

2. List available community strings.

```
~(keystone_admin)$ system snmp-comm-list
+----------------+--------------------+--------+
```

```
| SNMP community | View                | Access |
+----------------+---------------------+--------+
| commstr1       | .1                  | ro     |
+----------------+---------------------+--------+
```

3. Query details of a specific community string.

```
~(keystone_admin)$ system snmp-comm-show commstr1
+------------+--------------------------------------+
| Property   | Value                                |
+------------+--------------------------------------+
| access     | ro                                   |
| created_at | 2014-08-14T21:12:10.037637+00:00     |
| uuid       | eccf5729-e400-4305-82e2-bdf344eb868d |
| community  | commstr1                             |
| view       | .1                                   |
+------------+--------------------------------------+
```

4. Delete a community string.

```
~(keystone_admin)$ system snmp-comm-delete commstr1
Deleted community commstr1
```

Community strings in HP Helion OpenStack Carrier Grade provide query access to any SNMP monitor workstation that can reach the controller's OAM address on UDP port 161.

You can verify SNMP access using any monitor tool. For example, the freely available command `snmpwalk` can be issued from any host to list the state of all SNMP Object Identifiers (OID):

```
$ snmpwalk -v 2c -c commstr1 10.10.10.100 > oids.txt
```

In this example, 10.10.10.100 is the HP Helion OpenStack Carrier Grade OAM floating IP address. The output, which is a large file, is redirected to the file `oids.txt`.

Configuring SNMP Trap Destinations

SNMP trap destinations are hosts configured in HP Helion OpenStack Carrier Grade to receive unsolicited SNMP notifications.

Destination hosts are specified by IP address, or by host name if it can be properly resolved by HP Helion OpenStack Carrier Grade. Notifications are sent to the hosts using a designated community string so that they can be validated.

1. Configure IP address 10.10.10.1 to receive SNMP notifications using the community string *commstr1*.

```
~(keystone_admin)$ system snmp-trapdest-add -c commstr1 --ip_address
 10.10.10.1
+------------+--------------------------------------+
| Property   | Value                                |
+------------+--------------------------------------+
| uuid       | c7b6774e-7f45-40f5-bcca-3668de2a186f |
| ip_address | 10.10.10.1                           |
| community  | commstr1                             |
| type       | snmpv2c_trap                         |
| port       | 162                                  |
| transport  | udp                                  |
+------------+--------------------------------------+
```

The following are attributes associated with the new community string:

uuid

The UUID associated with the trap destination object.

ip_address

The trap destination IP address.

community

The community string value to be associated with the notifications.

type

snmpv2c_trap, the only supported message type for SNMP traps.

port

The destination UDP port that SNMP notifications are sent to.

transport

The transport protocol used to send notifications.

2. List defined trap destinations.

```
~(keystone_admin)$ system snmp-trapdest-list
+------------+----------------+------+--------------+-----------+
| IP Address  | SNMP Community | Port | Type         | Transport |
+-------------+----------------+------+--------------+-----------+
| 10.10.10.1  | commstr1       | 162  | snmpv2c_trap | udp       |
+-------------+----------------+------+--------------+-----------+
```

3. Query access details of a specific trap destination.

```
~(keystone_admin)$ system snmp-trapdest-show 10.10.10.1
+------------+--------------------------------------+
| Property   | Value                                |
+------------+--------------------------------------+
| uuid       | c7b6774e-7f45-40f5-bcca-3668de2a186f |
| ip_address | 10.10.10.1                           |
| community  | commstr1                             |
| type       | snmpv2c_trap                         |
| port       | 162                                  |
| transport  | udp                                  |
+------------+--------------------------------------+
```

4. Disable the sending of SNMP notifications to a specific IP address.

```
~(keystone_admin)$ system snmp-trapdest-delete 10.10.10.1
Deleted ip 10.10.10.1
```

System Alarms CLI Commands

You can use the CLI to find information about currently active and previously triggered system alarms.

The following commands are used to interact with the alarms subsystem: `system alarm-list`, `system alarm-show`, `system alarm-delete`, and `system alarm-history-list`. Before using the commands you must log in to the active controller as the Keystone **admin** user. See *Linux User Accounts* on page 18 for details.

system alarm-list

The command `system alarm-list` lists currently active alarms, as illustrated below (the output is split in two pieces for presentation purposes only):

```
~(keystone_admin)$ system alarm-list

+--------------+----------+--------------------+----------+---------------
| UUID         | Alarm ID | Entity Instance ID | Severity |  Time Stamp
```

```
+--------------+----------+--------------------+----------+---------------
| 4ab5698a-... | 100.104  | host=controller-0  | critical |  2014-06-25...
+--------------+----------+--------------------+----------+---------------

--+-------------------------------------------------------------+
  | Reason Text                                                 |
--+-------------------------------------------------------------+
  | /dev/sda3 severity critical threshold set (0.00 MB left)    |
--+-------------------------------------------------------------+
```

This example lists a single critical alarm on host **controller-0** regarding running out of space on disk unit `/dev/sda3`. Each alarm object is listed with a unique UUID which you can use to obtain additional information.

Specific subsets of alarms, or a particular alarm, can be listed using one of the following --query command filters:

Query Filter	**Comment**
`uuid=<uuid>`	Query alarm by UUID, for example: `$ system alarm-list --query uuid=4ab5698a-19cb...`
`alarm_id=<alarm id>`	Query alarms by alarm ID, for example: `$ system alarm-list --query alarm_id=100.104`
`alarm_type=<type>`	Query alarms by type, for example: `$ system alarm-list --query \ alarm_type=operational-violation`
`entity_type_id=<type id>`	Query alarms by entity type ID, for example: `$ system alarm-list --query \ entity_type_id=system.host`
`entity_instance_id=<instance id>`	Query alarms by entity instance id, for example: `$ system alarm-list --query \ entity_instance_id=host=compute-0`
`severity=<severity>`	Query alarms by severity type, for example: `$ system alarm-list --query severity=warning` The valid severity types are *critical*, *major*, *minor*, and *warning*.

Query command filters can be combined into a single expression separated by semicolons, as illustrated in the following example:

```
$ system alarm-list -q
 'alarm_id=400.002;entity_instance_id=service_domain=controller.service_group=directory-
services'
```

system alarm-show

The command `system alarm-show` presents additional information about a currently active alarm, as illustrated below:

```
~(keystone_admin)$ system alarm-show 4ab5698a-19cb-4c17-bd63-302173fef62c

+------------------------+------------------------------------------------+
| Property               | Value                                          |
+------------------------+------------------------------------------------+
| alarm_id               | 100.104                                        |
| alarm_state            | set                                            |
| alarm_type             | operational-violation                          |
| entity_instance_id     | system=hp380-1_4.host=controller-0             |
| entity_type_id         | system.host                                    |
| probable_cause         | threshold-crossed                              |
| proposed_repair_action | /dev/sda3 check usage                          |
| reason_text            | /dev/sda3 critical threshold set (0.00 MB left) |
| service_affecting      | False                                          |
| severity               | critical                                       |
| suppression            | True                                           |
| timestamp              | 2014-06-25T16:58:57.324613                     |
| uuid                   | 4ab5698a-19cb-4c17-bd63-302173fef62c           |
+------------------------+------------------------------------------------+
```

The pair of attributes **(alarm_id, entity_instance_id)** uniquely identifies an active alarm:

alarm_id

An ID identifying the particular alarm condition. Note that there are some alarm conditions, such as *administratively locked* , that can be raised by more than one entity-instance-id.

entity_instance_id

Type and instance information of the object raising the alarm. A period-separated list of (key, value) pairs, representing the containment structure of the overall entity instance. This structure is used for processing hierarchical clearing of alarms.

system alarm-delete

The command `system alarm-delete` is used to manually delete an alarm that remains active for no apparent reason, which may happen in rare conditions. Alarms usually clear automatically when the trigger condition is corrected. Use this command as illustrated below:

```
~(keystone_admin)$ system alarm-delete 4ab5698a-19cb-4c17-bd63-302173fef62c
```

Manually deleting an alarm should not be done unless it is absolutely clear that there is no reason for the alarm to be active.

system alarm-history-list

The command `system alarm-history-list` is used to query the historical alarm table. It operates on an alarm ring buffer of up to 2000 entries used by the alarms subsystem to sequentially store active alarm change events. In its simplest form, without any parameters, the command returns a list of the 20 most recent change events in reverse chronological order, the most recent event first. Use the -l option to specify the size of the list. The following command lists the 30 more recent change events:

```
~(keystone_admin)$ system alarm-history-list -l 30
```

The console output is automatically paginated when the list size is greater than 20. Press the `Enter` key to go the next page, or press `q` to quit.

Specific alarms, or alarm subsets, in the ring buffer can be listed using the --query command filters accepted by the `system alarms-list` command. For example, use the following command to query alarm events in the ring buffer by type ID:

```
~(keystone_admin)$ system alarm-history-list --query alarm_id=100.104
```

Two additional command filters are available to restrict the command output to change events in a particular time slot, as follows:

Query Filter	**Comment**

`start=<time_stamp>`	Query change events that occurred at or after a particular time, for example:
	`$ system alarm-history-list --query \`
	`start=2014-11-26T18:58:53`

`end=<time_stamp>`	Query change events that occurred at or before a particular time, for example:
	`$ system alarm-history-list --query \`
	`end=2014-11-26T18:59:53`

Time stamps must be entered in a suitable ISO 8601 date and time format. Some examples are: `2014`, `2014-11-26`, `2014-11-28T16:39`, and `2014-11-28T16:42:35.647157`.

Query command filters can be combined into a single expression separated by semicolons, as illustrated in the following example:

```
~(keystone_admin)$ system alarm-history-list -l 10 \
-q 'start=2014-11-26T18:58:53;end=2014-11-26T18:59:53'
```

Alarms Reference Table

The system inventory and maintenance service reports system changes with different degrees of severity. Use the reported alarms to monitor the overall health of the system.

In the following tables the severity of the alarms is represented by one or more letters, as follows:

- C: Critical
- M: Major
- m: Minor
- W: Warning

A comma-separated list of letters is used when the alarm can be triggered with one of several severity levels.

Resource Alarms

Alarm ID	**Reason Text**	**Entity Instance ID**	**Severity**	**Proposed Repair Action**
100.101	Platform CPU threshold exceeded; threshold x%, actual y% .	host=<*hostname*>	C, M, m	Monitor and if condition persists, contact next level of support.

Alarm ID	Reason Text	Entity Instance ID	Severity	Proposed Repair Action
100.102	VSwitch CPU threshold exceeded; threshold x%, actual y% .	host=<*hostname*>	C, M, m	Monitor and if condition persists, contact next level of support.
100.103	Memory threshold exceeded; threshold x%, actual y% .	host=<*hostname*>	C, M, m	Monitor and if condition persists, contact next level of support; may require additional memory on Host.
100.104	File System threshold exceeded; threshold x%, actual y% .	host=<*hostname*>.filesystem=<mount-dir> OR filesystem=<mount-dir>	C, M, m	Monitor and if condition persists, contact next level of support.
		host=<*hostname*>.volumegroup=<volumegroup-name>	M, m	Monitor and if condition persists, consider adding additional physical volumes to the volume group.
100.105	No access to remote VM volumes.	host=<*hostname*>	M	Check Management and Infrastructure Networks and Controller or Storage Nodes.
100.106	'OAM' Port failed.	host=<*hostname*>.port=<port-name>	M	Check cabling and far-end port configuration and status on adjacent equipment.
100.107	'OAM' Interface degraded. OR 'OAM' Interface failed.	host=<*hostname*>.interface=<if-name>	C, M	Check cabling and far-end port configuration and status on adjacent equipment.
100.108	'MGMT' Port failed.	host=<*hostname*>.port=<port-name>	M	Check cabling and far-end port configuration and status on adjacent equipment.
100.109	'MGMT' Interface degraded. OR 'MGMT' Interface failed.	host=<*hostname*>.interface=<if-name>	C, M	Check cabling and far-end port configuration and status on adjacent equipment.
100.110	'INFRA' Port failed.	host=<*hostname*>.port=<port-name>	M	Check cabling and far-end port configuration and status on adjacent equipment.
100.111	'INFRA' Interface degraded. OR 'INFRA' Interface failed.	host=<*hostname*>.interface=<if-name>	C, M	Check cabling and far-end port configuration and status on adjacent equipment.

Maintenance Alarms

Alarm ID	Reason Text	Entity Instance ID	Severity	Proposed Repair Action
200.001	Host was administratively locked to take it out-of-service.	host=<*hostname*>	W	Administratively unlock Host to bring it back in-service.
200.004	Host experienced a service-affecting failure. Resetting Host.	host=<*hostname*>	C	If problem consistently occurs after Host is reset, contact next level of support or lock and replace failing host.
200.005	Degrade: Host is experiencing an intermittent 'Management Network' communication failure that has exceeded its lower alarming threshold. Failure: Host is experiencing a persistent critical 'Management Network' communication failure. Resetting Host.	host=<*hostname*>	C, M	If problem consistently occurs after Host is reset, contact next level of support or lock and replace failing host.
200.009	Degrade: Host is experiencing an intermittent 'Infrastructure Network' communication failure that has exceeded its lower alarming threshold. Failure: Host is experiencing a persistent critical 'Infrastructure Network' communication failure. Resetting Host.	host=<*hostname*>	C, M	If problem consistently occurs after Host is reset, contact next level of support or lock and replace failing host.
200.006	One or more Critical:'<process list>' and/or Degraded:'<process list>' processes on Host	host=<*hostname*>	C, M	If problem consistently occurs after Host is reset, contact next level of support or lock and replace failing host.

Alarm ID	Reason Text	Entity Instance ID	Severity	Proposed Repair Action
	have failed and can not be recovered.			
200.007	Critical: (with host degrade): Host is degraded due to a 'critical' out-of-tolerance reading from the '<sensorname>' sensor Major: (with host degrade) Host is degraded due to a 'major' out-of-tolerance reading from the '<sensorname>' sensor Minor: Host is reporting a 'minor' out-of-tolerance reading from the '<sensorname>' sensor	host=<*hostname*>.sensor=<sensorname>	C, M, m	If problem consistently occurs after Host is power cycled and or reset, contact next level of support or lock and replace failing host.
200.008	ntpd' process has failed on Host.	host=<*hostname*>	m	'ntpd' is a process that can not be auto recovered. The Host must be re-enabled (locked and then unlocked) to clear this alarm. If the alarm continues to persist then contact next level of support to investigate and recover.
200.010	Access to board management module has failed.	host=<*hostname*>	W	Check Host's board management configuration and connectivity.
200.011	Host encountered a critical configuration failure during initialization. Resetting Host.	host=<*hostname*>	C	If problem consistently occurs after Host is reset, contact next level of support or lock and replace failing host.
200.0112	In-Service failure of host's controller function while compute services remain healthy.	host=<*hostname*>	M	Lock and then Unlock host to recover. Avoid using 'Force Lock' action as that will impact compute services running on this host. If lock action fails then contact next level of support to investigate and recover.

Alarm ID	Reason Text	Entity Instance ID	Severity	Proposed Repair Action
200.013	In-Service failure of host's compute function on host with only available and healthy controller service.	host=<*hostname*>	M	Enable second controller as soon as possible and then optionally Lock and Unlock host to recover local compute services on this host.
200.014	The Hardware Monitor was unable to load, configure and monitor one or more hardware sensors.	host=<*hostname*>	m	Check Board Management Controller provisioning. Try reprovisioning the BMC. If problem persists try power cycling the host and then the entire server including the BMC power. If problem persists then contact next level of support.

Storage Alarms

Alarm ID	Reason Text	Entity Instance ID	Severity	Proposed Repair Action
800.001	Storage Alarm Condition: 1 mons down, quorum 1,2 controller-1,storage-0	cluster=<*dist-fs-uuid*>	C, M	If problem persists, contact next level of support.

Data Networking Alarms

Alarm ID	Reason Text	Entity Instance ID	Severity	Proposed Repair Action
300.001	'Data' Port failed.	host=<*hostname*>.port=<port-uuid>	M	Check cabling and far-end port configuration and status on adjacent equipment.
300.002	'Data' Interface degraded. OR 'Data' Interface failed.	host=<*hostname*>.interface=<if-uuid>	M, C	Check cabling and far-end port configuration and status on adjacent equipment.
300.003	Networking Agent not responding.	host=<*hostname*>.agent=<agent-uuid>	M	If condition persists, attempt to clear issue by administratively locking and unlocking the Host.
300.004	No enabled compute host with connectivity to provider network.	host=<*hostname*>.providernet=<pnet-uuid>	M	Enable compute hosts with required provider network connectivity.

Controller HA Alarms

Alarm ID	Reason Text	Entity Instance ID	Severity	Proposed Repair Action
400.001	Service group failure; <list of affected services>. OR Service group degraded; <list of affected services>. OR Service group warning; <list of affected services>.	service_domain=<domain_name>,service_group=<group_name>,host=<*hostname*>	C, M, m	Contact next level of support.
400.002	Service group loss of redundancy; expected <num> standby member<s> but only <num> standby member<s> available. OR Service group loss of redundancy; expected <num> standby member<s> but only <num> standby member<s> available. OR Service group loss of redundancy; expected <num> active member<s> but no active members available. OR Service group loss of redundancy; expected <num> active member<s> but only <num> active member<s> available.	service_domain=<domain_name>.service_group=<group_name>	M	Bring group <group_name> back in to service, otherwise contact next level of support.
400.003	License key has expired or is invalid; a valid license key is required for operation. OR Evaluation license key will expire on <date>; there are <num_days> days remaining in this evaluation. OR	host=<*hostname*>	C	Contact next level of support to obtain a new license key.

Alarm ID	Reason Text	Entity Instance ID	Severity	Proposed Repair Action
	Evaluation license key will expire on <date>; there is only 1 day remaining in this evaluation.			
400.004	Service group software modification detected; <list of affected files>.	host=<*hostname*>	M	Contact next level of support.
400.005	Communication failure detected with peer over port <linux-ifname>. OR Communication failure detected with peer over port <linux-ifname> within the last 30 seconds.	host=<*hostname*>.network=<mgmt | oam | infra>	M	Check cabling and far-end port configuration and status on adjacent equipment.

Backup and Restore Alarms

Alarm ID	Reason Text	Entity Instance ID	Severity	Proposed Repair Action
210.001	System Backup in progress.	host=controller	m	No action required.

System Configuration

Alarm ID	Reason Text	Entity Instance ID	Severity	Proposed Repair Action
250.001	<*hostname*> Configuation is out-of-date.	host=<*hostname*>	M	Administratively lock and unlock <*hostname*> to update config.
250.01	<*hostname*> Provisioning compute required. (This alarm only applies in the 2-Server/Combined load).	host=<*hostname*>	M	Administratively lock and unlock <*hostname*> to update provisioning of Coimpute functionality.

Software Management Alarms

Alarm ID	Reason Text	Entity Instance ID	Severity	Proposed Repair Action
900.001	Patching operation in progress.	host=controller	m	Complete reboots of affected hosts.
900.002	Obsolete patch in system.	host=controller	W	Remove and delete obsolete patches.

Alarm ID	Reason Text	Entity Instance ID	Severity	Proposed Repair Action
900.003	Patch host install failure.	host=<*hostname*>	M	Undo patching operation.

Virtual Machine Instance Alarms

Alarm ID	Reason Text	Entity Instance ID	Severity	Proposed Repair Action
700.003	Instance <*instance-name*> is failed	instance=<*instance_uuid*>	C	The system will attempt recovery; no repair action required
700.007	Instance <*instance-name*> is paused	instance=<*instance_uuid*>	C	Unpause the instance
700.009	Instance <*instance-name*> is suspended	instance=<*instance_uuid*>	C	Resume the instance
700.012	Instance <*instance-name*> is live migrating	instance=<*instance_uuid*>	W	Wait for live migration to complete; if problem persists contact next level of support
700.013	Instance <*instance-name*> is cold migrating	instance=<*instance_uuid*>	C	Wait for cold migration to complete; if problem persists contact next level of support
700.014	Instance <*instance-name*> has been cold-migrated	instance=<*instance_uuid*>	C	Confirm or revert cold-migrate of instance
700.017	Instance <*instance-name*> is evacuating	instance=<*instance_uuid*>	C	Wait for evacuate to complete; if problem persists contact next level of support
700.020	Instance <*instance-name*> is stopped	instance=<*instance_uuid*>	C	Start the instance
700.021	Instance <*instance-name*> is rebooting	instance=<*instance_uuid*>	C	Wait for reboot to complete; if problem persists contact next level of support
700.022	Instance <*instance-name*> is rebuilding	instance=<*instance_uuid*>	C	Wait for rebuild to complete; if problem persists contact next level of support
700.023	Instance <*instance-name*> is resizing	instance=<*instance_uuid*>	C	Wait for resize to complete; if problem persists contact next level of support
700.024	Instance <*instance-name*> has been resized	instance=<*instance_uuid*>	C	Confirm or revert resize of instance

Alarm ID	Reason Text	Entity Instance ID	Severity	Proposed Repair Action
700.027	Guest Heartbeat not established for instance <*instance-name*>	instance=<*instance_uuid*>	M	Verify that the instance is running the Guest-Client daemon, or disable Guest Heartbeat for the instance if no longer needed, otherwise contact next level of support

Customer Logs

You can obtain information about transient events using Customer Logs.

Certain system events that do not result in node state changes and typically do not require immediate customer action, such as instance deletions or failed migration attempts, are recorded in Customer Logs. Each log describes a single event. The logs are held in a buffer, with older logs discarded as needed to release logging space.

The logs are displayed in a list, along with summary information. For each individual log, you can view detailed information. For more information, see *Viewing Customer Logs* on page 187.

Viewing Customer Logs

You can use the web administration interface or the CLI to examine customer logs.

To view customer logs, you must be logged in with administrative privileges.

You can use the command-line interface to view customer logs by acquiring Keystone credentials and using the `system log-list` and `system log-show` commands. Alternatively, you can use the web administration interface using the steps given later in this section.

- To acquire Keystone **admin** credentials, type the following command:

```
$ source /etc/nova/openrc
```

- To list logs, use the following command:

```
~(keystone_admin)$ system log-list [-q query] [-l limit]
```

where

query

is a list of one or more key-value pairs with comparison operators, separated by semicolons.

limit

is the maximum number of logs to return, beginning with the most recent.

For example:

```
~(keystone_admin)$ system log-list -q severity=critical -l 20
```

returns the 20 most recent logs with **critical** severity.

- To view details for an individual log, use the following command:

```
~(keystone_admin)$ system log-show uuid
```

where *uuid* is the unique identifier for the log (obtained using the `system log-list` command).

1. In the main menu, select **Admin** > **System** > **Fault Management**.

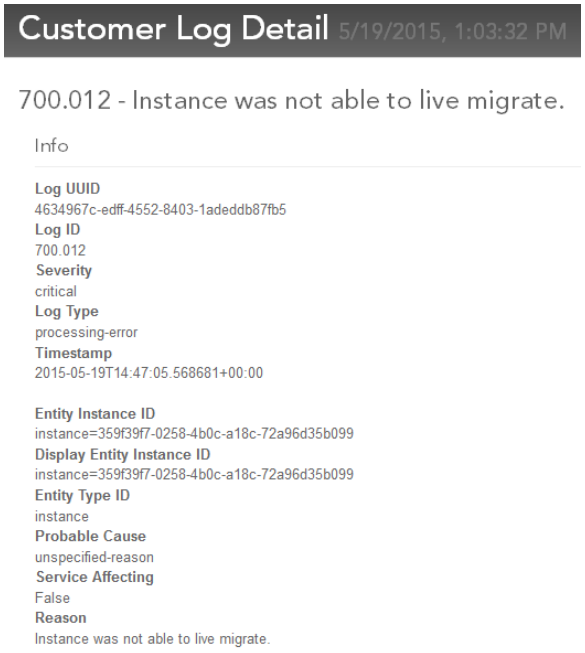
2. On the Fault Management page, select the **Customer Logs** tab.

   The logs are displayed in chronological order, beginning with the most recent.

   By default, 20 items are shown per page. To see older items, scroll to the bottom of the list and click **Next**. You can change the number of items displayed per page using the **Default Limit** drop-down list.

   You can also filter the items using the **Filter** text box. Type the filter string, and then click the magnifying-glass icon. The results include only those logs that contain the string in the Customer Log Detail.

3. To see the Customer Log Detail, click the **Timestamp** identifier for the log.



Customer Logs Reference Table

The Customer Logs include events that do not require immediate user action.

The following types of events are included in the Customer Logs. The severity of the events is represented in the table by one or more letters, as follows:

- C: Critical
- M: Major
- m: Minor
- W: Warning

Log ID	Reason	Entity Instance ID	Severity
700.001	Instance <*instance-name*> is enabled	instance=<*instance_uuid*>	C

Log ID	Reason	Entity Instance ID	Severity
700.002	Instance <*instance-name*> has crashed	instance=<*instance_uuid*>	C
700.003	Instance <*instance-name*> has failed	instance=<*instance_uuid*>	C
700.004	Deleting instance <*instance-name*>	instance=<*instance_uuid*>	C
700.005	Deleted instance <*instance-name*>	instance=<*instance_uuid*>	C
700.006	Pause issued against instance <*instance-name*>	instance=<*instance_uuid*>	C
700.006	Pause rejected for instance <*instance-name*>, reason=<*reason-text*>	instance=<*instance_uuid*>	C
700.007	Instance <*instance-name*> is paused	instance=<*instance_uuid*>	C
700.008	Unpause issued against instance <*instance-name*>	instance=<*instance_uuid*>	C
700.008	Unpause rejected for instance <*instance-name*>, reason=<*reason-text*>	instance=<*instance_uuid*>	C
700.009	Suspend issued against instance <*instance-name*>	instance=<*instance_uuid*>	C
700.009	Suspend rejected for instance <*instance-name*>, reason=<*reason-text*>	instance=<*instance_uuid*>	C
700.010	Instance <*instance-name*> is supended	instance=<*instance_uuid*>	C
700.011	Resume issued against instance <*instance-name*>	instance=<*instance_uuid*>	C
700.011	Resume rejected for instance <*instance-name*>, reason=<*reason-text*>	instance=<*instance_uuid*>	C
700.012	Live-Migrate issued against instance <*instance-name*>	instance=<*instance_uuid*>	C
700.012	Live-Migrate rejected for instance <*instance-name*>, reason=<*reason-text*>	instance=<*instance_uuid*>	C

Log ID	Reason	Entity Instance ID	Severity
700.012	Live-Migrate failed for instance <*instance-name*>	instance=<*instance_uuid*>	C
700.012	Live-Migrate inprogress for instance <*instance-name*>	instance=<*instance_uuid*>	C
700.012	Live-Migrate aborted for instance <*instance-name*>, reason=<*reason for abort*>	instance=<*instance_uuid*>	C
700.013	Cold-Migrate issued against instance <*instance-name*>	instance=<*instance_uuid*>	C
700.013	Cold-Migrate rejected for instance <*instance-name*>, reason=<*reason-text*>	instance=<*instance_uuid*>	C
700.013	Cold-Migrate failed for instance <*instance-name*>	instance=<*instance_uuid*>	C
700.013	Cold-Migrate inprogress for instance <*instance-name*>	instance=<*instance_uuid*>	C
700.014	Instance <*instance-name*> has been cold-migrated	instance=<*instance_uuid*>	C
700.015	Cold-Migrate-Confirm issued against instance <*instance-name*>	instance=<*instance_uuid*>	C
700.015	Cold-Migrate-Confirm rejected for instance <*instance-name*>, reason=<*reason-text*>	instance=<*instance_uuid*>	C
700.016	Cold-Migrate-Revert issued against instance <*instance-name*>	instance=<*instance_uuid*>	C
700.016	Cold-Migrate-Revert rejected for instance <*instance-name*>, reason=<*reason-text*>	instance=<*instance_uuid*>	C
700.017	Evacuate issued against instance <*instance-name*>	instance=<*instance_uuid*>	C
700.017	Evacuate rejected for instance <*instance-name*>, reason=<*reason-text*>	instance=<*instance_uuid*>	C
700.017	Evacuating instance <*instance-name*>	instance=<*instance_uuid*>	C

Log ID	Reason	Entity Instance ID	Severity
700.018	Start issued against instance <*instance-name*>	instance=<*instance_uuid*>	C
700.018	Start rejected for instance <*instance-name*>, reason=<*reason-text*>	instance=<*instance_uuid*>	C
700.019	Stop issued against instance <*instance-name*>	instance=<*instance_uuid*>	C
700.019	Stop rejected for instance <*instance-name*>, reason=<*reason-text*>	instance=<*instance_uuid*>	C
700.020	Instance <*instance-name*> is stopped	instance=<*instance_uuid*>	C
700.021	Reboot issued against instance <*instance-name*>, soft-reboot	instance=<*instance_uuid*>	C
700.021	Reboot issued against instance <*instance-name*>, hard-reboot	instance=<*instance_uuid*>	C
700.021	Reboot rejected for instance <*instance-name*>, reason=<*reason-text*>	instance=<*instance_uuid*>	C
700.021	Rebooting instance <*instance-name*>	instance=<*instance_uuid*>	C
700.022	Rebuild issued against instance <*instance-name*>	instance=<*instance_uuid*>	C
700.022	Rebuild rejected for instance <*instance-name*>, reason=<*reason-text*>	instance=<*instance_uuid*>	C
700.022	Rebuilding instance <*instance-name*>	instance=<*instance_uuid*>	C
700.023	Resize issued against instance <*instance-name*>	instance=<*instance_uuid*>	C
700.023	Resize rejected for instance <*instance-name*>, reason=<*reason-text*>	instance=<*instance_uuid*>	C
700.023	Resizing instance <*instance-name*>	instance=<*instance_uuid*>	C
700.024	Instance <*instance-name*> has been resized	instance=<*instance_uuid*>	C

Log ID	Reason	Entity Instance ID	Severity
700.025	>Resize-Confirm issued against instance <*instance-name*>	instance=<*instance_uuid*>	C
700.025	Resize-Confirm rejected for instance <*instance-name*>, reason=<*reason-text*>	instance=<*instance_uuid*>	C
700.026	Resize-Revert issued against instance <*instance-name*>	instance=<*instance_uuid*>	C
700.026	Resize-Revert rejected for instance <*instance-name*>, reason=<*reason-text*>	instance=<*instance_uuid*>	C
700.027	Guest Heartbeat established for instance <*instance-name*>	instance=<*instance_uuid*>	M
700.027	Guest Heartbeat disconnected for instance <*instance-name*>	instance=<*instance_uuid*>	M
700.027	Guest Heartbeat failed for instance <*instance-name*>	instance=<*instance_uuid*>	C
700.027	Guest Heartbeat failed for instance <*instance-name*>, soft-reboot repair action requested	instance=<*instance_uuid*>	C
700.027	Guest Heartbeat failed for instance <*instance-name*>, stop repair action requested	instance=<*instance_uuid*>	C

Chapter 7

Managing Software Patches

Topics:

Patches to the system software become available as needed to address issues associated with a current HP Helion OpenStack Carrier Grade software release. They must be uploaded to the active controller and applied to all required hosts in the cluster.

The patching system is implemented using the following components:

Patch Controller

This is a daemon running on the active controller. The daemon queries the patch agents to collect information about the state of patches on every host in the cluster. It also communicates with the patch agents to command patch operations to take place.

Patch Agent

These are daemons running on the cluster hosts, one agent per host. Each patch agent maintains up-to-date information about the status of patches on the host, and responds to queries and commands from the patch controller.

Patching Commands

The `sw-patch` command is available on the active controller. It provides the user interface to process the patches, including querying the state of a patch, listing affected hosts, and applying, installing, and removing patches.

The `system host-patch-reboot` command provides a convenient way to lock a host, install all pending patches, reboot, and bring the host back to the unlock-available state.

Patch Storage Area

A central storage area maintained by the patch controller. Patches are initially uploaded to the patch storage area and remain there until they are deleted.

Software Updates Repository

A central repository of software updates associated with patches applied to the system. This repository is used by all hosts in the cluster to identify the software updates and rollbacks required on each host.

The overall flow for installing a patch from the command line interface on a working HP Helion OpenStack Carrier Grade cluster is the following:

1. Download the patch servers to a workstation that can reach the active controller through the OAM network.
2. Copy the patch to the active controller using the cluster's OAM floating IP address as the destination point. You can use a command such as `scp`

to copy the patch. The patching workflows presented in this document assume that this step is complete already, that is, they assume that the patch is already available from the file system of the active controller.

3. Upload the new patch to the patching storage area. This step makes the new patch available within the patching system, but does not install it to the cluster yet. For all purposes, the patch is dormant.
4. Apply the patch, and install it on each of the affected hosts in the cluster.

Patching the system can be done using the web administration interface or the command line interface on the active controller. When using the web administration interface you upload the patch directly from your workstation using a file browser window provided by the patch upload facility.

A special case occurs during the initial provisioning of a cluster, when you want to patch **controller-0** before the system software is configured. This can only be done from the command line interface. See *Installing Patches Locally* on page 200 for details.

The following sections introduce the core patching concepts using the command line interface. It is therefore advised that you read this content before reading the section *Patching Using the Web Administration Interface* on page 202, which assumes you are already familiar with the patching workflow.

Populating the Patch Storage Area

Patches have to be uploaded to the HP Helion OpenStack Carrier Grade patch storage area before they can be applied.

1. Log in as Keystone user **admin** to the active controller.

   Log in as user **wrsroot** to the active controller and source the script `/etc/nova/openrc` to obtain administrative privileges as described in *Linux User Accounts* on page 18.

2. Upload the patch file to the patch storage area.

```
~(keystone_admin)$ sudo sw-patch upload /tmp/patches/
CONTROLLER_PATCH_0001.patch
CONTROLLER_PATCH_0001 is now available
```

   This example uploads a single patch to the storage area. You can specify multiple patch files on the same command separating their names with spaces.

   Alternatively, you can upload all patch files stored in a directory using a single command, as illustrated in the following example:

```
~(keystone_admin)$ sudo sw-patch upload-dir /tmp/patches
```

   The patch files are available now in the patch storage area, but they have not yet been applied to the cluster.

3. Verify the status of the patch.

```
~(keystone_admin)$ sudo sw-patch query
       Patch ID             Patch State
========================    ===========
PATCH_0001   Available
```

   The patch state is *Available* now, indicating that it is included in the patch storage area. Further details about the patch can be retrieved as follows:

```
~(keystone_admin)$ sudo sw-patch show PATCH_0001
PATCH_0001:
    Patch State:    Available
    Release Status: Released
    Summary:        Add logs to hbsClient and libalarm-clock
    Description:    Modified log in hbsClient. Expect to see this log on
                    all blade types  'hbsClient patches by
                    PATCH_0001' when hbsClient
                    restarts. Also added a log to ac_init() function
                    that reads "PATCH_0001 patched
                    libalarm-clock".  Should be seen when heartbeat
                    process is restarted on controller and compute.
    Install Instructions:
                    install text here
    Requires:
                    PATCH_0001
    Contents:
                    mtce-common-1.0-r23.x86_64.rpm
                    mtce-common-dev-1.0-r23.x86_64.rpm
                    mtce-common-pmon-1.0-r23.x86_64.rpm
                    mtce-common-rmon-1.0-r23.x86_64.rpm
                    libalarm-clock-1.0-r1.x86_64.rpm
```

4. Delete a patch from the patch storage area.

Patches in the *Available* state can be deleted as illustrated in the following command:

```
~(keystone_admin)$ sudo sw-patch delete PATCH_0001
```

The patch to delete from the patch storage area is identified by the patch ID reported by the `sw-patch query` command. You can provide multiple patch IDs to the delete command separated by spaces.

Installing Patches

After adding a patch to the patch storage area, you must move it to the software updates repository, which manages patch distribution for the cluster. From there, you can install patches to the hosts that require them.

Before you can install a patch, you must make it *available* by adding it to the patch storage area. For more information, see *Populating the Patch Storage Area* on page 195.

Some of the available patches may be required on controller hosts only, while others may be required on compute or storage hosts. When you *apply* an available patch, it is added to the software updates repository, ready for installation on any host type that requires it. You can then install all applicable patches to each host in turn.

To keep track of patch installation, you can use the `sw-patch query` command.

```
~(keystone_admin)$ sudo sw-patch query
       Patch ID             Patch State
========================    ===========
PATCH_0001   Available
PATCH_0002   Available
PATCH_0003   Available
```

This shows the **Patch State** for each of the patches in the patch storage area:

Available

A patch in the *Available* state has been added to the patch storage area, but has not yet been added to the software updates repository or installed on the hosts.

Partial-Apply

A patch in the *Partial-Apply* state has been added to the software updates repository using the `sw-patch apply` command, but has not been installed on all hosts that require it. It may have been installed on some but not others, or it may not have been installed on any hosts.

Applied

A patch in the *Applied* state has been installed on all hosts that require it, using either the `system host-patch-reboot` command (for use with a locked host) or the `sw-patch host-install` command (for use with an unlocked host, for example during controller configuration).

You can use the `sw-patch query-hosts` command to see which hosts are fully patched (**Patch Current**). This also shows which hosts require reboot, either because they are not fully patched, or because they are fully patched but need restarting to make the patches take effect.

```
~(keystone_admin)$ sudo sw-patch query-hosts
  Hostname       IP Address    Patch Current  Reboot Required
============  ==============  =============  ===============
compute-0     192.168.204.15       Yes             No
controller-0  192.168.204.3        No              Yes
controller-1  192.168.204.4        No              Yes
```

The following example illustrates the process of installing a single patch on the cluster. It assumes that the patch is in the *Available* state already, as described in *Populating the Patch Storage Area* on page 195, and that only the controller nodes must be patched.

1. Log in as Keystone user **admin** to the active controller.

   Log in as user **wrsroot** to the active controller and source the script `/etc/nova/openrc` to obtain administrative privileges as described in *Linux User Accounts* on page 18.
2. Verify that the patches are available using the command `sw-patch query`.

```
~(keystone_admin)$ sudo sw-patch query
       Patch ID             Patch State
========================   ===========
PATCH_0001   Available
PATCH_0002   Available
PATCH_0003   Available
```

3. Apply the patch.

```
~(keystone_admin)$ sudo sw-patch apply PATCH_0001
PATCH_0001 is now in the repo
```

   The patch is now in the *Partial-Apply* state, ready for installation from the software updates repository on the impacted hosts.
4. Optional: Apply all available patches in a single operation.

```
~(keystone_admin)$ sudo sw-patch apply --all
PATCH_0001 is now in the repo
PATCH_0002 is now in the repo
PATCH_0003 is now in the repo
```

   In this example there are three patches ready now for installation from the software updates repository.
5. Query the patching status of all hosts in the cluster.

   You can query the patching status of all hosts at any time as illustrated below. Note that the reported status is the accumulated result of all applied and removed patches in the software updates repository, and not just the status due to a particular patch.

```
~(keystone_admin)$ sudo sw-patch query-hosts
  Hostname      IP Address    Patch Current  Reboot Required
============  ==============  =============  ===============
compute-0     192.168.204.15       Yes             No
controller-0  192.168.204.3        No              Yes
controller-1  192.168.204.4        No              Yes
```

   For each host in the cluster, the following patch status fields are displayed:

   Patch Current

   Indicates whether there are patches pending for installation or removal on the host or not. If *Yes*, then all relevant patches in the software updates repository have been installed on, or removed from, the host already. If *No*, then there is at least one patch in either the *Partial-Apply* or *Partial-Remove* states that has not been applied to the host.

   Reboot Required

   Indicates whether the host has to be rebooted or not as a result of one or more patches that have been either applied or removed, or because it is not patch current.

   In this example, **compute-0** is up to date, no patches need to be installed and no reboot is required. By contrast, the controllers are not patch current, and therefore a reboot is required to install the patch.
6. Install all pending patches on **controller-0**.

a) Swact controller services.

```
~(keystone_admin)$ system host-swact controller-0
```

Before patching a controller node, you must transfer any active services running on the host to the other controller. Only then it is safe to lock the host.

b) Run the patching sequence.

The patching sequence locks the host, installs the patches, and forces a host reboot.

```
~(keystone_admin)$ system host-patch-reboot controller-0
host-patch-reboot controller-0 may take several minutes to complete.

2014-10-17-20-39-59_host-patch-reboot start.
...

2014-10-17-20-39-59_host-patch-reboot host-lock.
waiting for host controller-0 to transition to locked-disabled-
online ...
waiting for host controller-0 to transition to locked-disabled-
online ...
2014-10-17-20-40-14_Host controller-0 is locked-disabled-online.

2014-10-17-20-40-14_Patch install request
Installing patch for host controller-0 ...

2014-10-17-20-40-16_Patch install response
Patch response Info: Patch installation was successful on controller-0.

2014-10-17-20-40-16_host-patch-reboot host-unlock
Restoring initial administrative state: unlocked. Unlocking host
 controller-0 to return to initial administrative state.
```

All patches are now installed on **controller-0**. Querying the current patch status displays the following information:

```
~(keystone_admin)$ sudo sw-patch query-hosts
  Hostname      IP Address    Patch Current  Reboot Required
============  ==============  =============  ===============
compute-0     192.168.204.15       Yes             No
controller-0  192.168.204.3        Yes             No
controller-1  192.168.204.4        No              Yes
```

7. Optional: Manually install all pending patches on **controller-0**.

You may want to have finer control of the patching steps executed by the `system host-patch-reboot` command to determine when to install the patches and unlock the host.

a) Swact controller services.

```
~(keystone_admin)$ system host-swact controller-0
```

Before patching a controller node, you must transfer any active services running on the host to the other controller. Only then it is safe to lock the host.

b) Lock the host.

```
~(keystone_admin)$ system host-lock controller-0
```

You must lock the target host, controller, compute, or storage, before installing patches.

c) Install the pending patches.

```
~(keystone_admin)$ sudo sw-patch host-install controller-0
Patch installation was successful on controller-0
```

All patches are now installed on **controller-0**. Querying the current patching status displays the following information:

```
~(keystone_admin)$ sudo sw-patch query-hosts
  Hostname      IP Address    Patch Current  Reboot Required
============  ==============  =============  ===============
compute-0     192.168.204.15       Yes             No
controller-0  192.168.204.3        Yes             Yes
controller-1  192.168.204.4        No              Yes
```

For illustration purposes in this section, the command `sw-patch host-install` is presented as the tool to *install* patches. But in fact, this command also *removes* patches as well, as described in *Removing Patches* on page 201. The command is better described then as the tool to install and remove all patches in the *Partial-Apply* and *Partial-Remove* states on a host.

d) Unlock the target host.

```
~(keystone_admin)$ system host-unlock controller-0
```

Unlocking the host forces a reset of the host followed by a reboot. This ensures that the host is restarted in a known state.

All patches are now installed on **controller-0**.

8. Install all pending patches on **controller-1**.

Repeat the previous step targeting **controller-1** this time.

All patches are now installed on **controller-1** as well. Querying the current patching status displays the following information:

```
~(keystone_admin)$ sudo sw-patch query-hosts
  Hostname      IP Address    Patch Current  Reboot Required
============  ==============  =============  ===============
compute-0     192.168.204.15       Yes             No
controller-0  192.168.204.3        Yes             No
controller-1  192.168.204.4        Yes             No
```

9. Install any pending patches for the compute or storage hosts.

If the **Patch Current** status for a compute or storage host is **No**, apply the pending patches using the following command:

```
~(keystone_admin)$ system host-patch-reboot hostname
```

where *hostname* is the name of the host (for example, **compute-0**). This applies the patch and then reboots the host.

10. Confirm that all patches are installed and the HP Helion OpenStack Carrier Grade cluster is up-to-date.

Use the `sw-patch query` command to verify that all patches are **Applied**.

```
~(keystone_admin)$ sudo sw-patch query
       Patch ID             Patch State
========================   ===========
PATCH_0001   Applied
```

If the **Patch State** for any patch is still shown as **Available** or **Partial-Apply**, use the **sw-patch query-hosts** command to identify which hosts are not **Patch Current**, and then apply patches to them as described in the preceding steps.

The cluster is up to date now. All patches are installed.

Installing Patches Locally

When installing the HP Helion OpenStack Carrier Grade software on new hardware, it is recommended that you install the latest available patches on **controller-0** before running the `config_controller` script, and before installing the software on other hosts. This ensures that:

- The software on **controller-0**, and all other hosts, is up to date when the cluster comes alive.
- You reduce installation time by avoiding patching the system right after an out-of-date software installation is complete.

This exercise assumes that the patches to install are available on a USB flash drive, or from a server reachable by **controller-0**.

1. Initialize **controller-0**.

   Use the HP Helion OpenStack Carrier Grade bootable ISO image to initialize **controller-0**. See the *HP Helion OpenStack Carrier Grade Software Installation Guide* for details.

   This step takes you to the point where you use the console port to log in to **controller-0** as user **wrsroot**.

2. Copy the patches to the local file system.

   If the patches are on a USB flash drive, copy them from the appropriate mount point to some directory in the local file system. For example:

   ```
   $ mkdir /tmp/patches
   $ cp /media/wrsroot/data/CONTROLLER_PATCH_0001.patch /tmp/patches
   ```

   If the patches are available from a remote server, you need to manually initialize a suitable Ethernet interface first, and then transfer them to the local file system. For example:

   ```
   $ mkdir /tmp/patches
   $ sudo ifconfig eth3 192.168.1.100 netmask 255.255.255.0 up
   $ scp \
   me@192.168.1.150:/usr/local/patches/CONTROLLER_PATCH_0001.patch \
   /tmp/patches
   ```

   All patches are now in the directory `/tmp/patches`.

3. Populate the patch storage area.

   Upload the patches using the command `sw-patch upload` or `sw-patch upload-dir` as described in *Populating the Patch Storage Area* on page 195.

4. Apply the patches.

   Apply the patches using the command `sw-patch apply --all`.

   The patches are now in the software updates repository, ready to be installed.

5. Install the patches locally.

   ```
   $ sudo sw-patch install-local
   Patch installation is complete.
   Please reboot before continuing with configuration.
   ```

   This command installs all applied patches on **controller-0**.

6. Reboot **controller-0**.

   You must reboot the controller to ensure that it is running with the software fully patched.

7. Continue the installation.

   Log in again as **wrsroot** and continue with the software installation by running the `config_controller` script. See the *HP Helion OpenStack Carrier Grade Software Installation Guide* for further details and directions.

Once all hosts in the cluster are initialized, they are all running fully patched software. The HP Helion OpenStack Carrier Grade cluster is up to date.

Removing Patches

Patches in the *Applied* or *Partial-Apply* states can be removed if necessary, for example, when they trigger undesired or unplanned effects on the cluster.

Rolling back patches is conceptually identical to installing patches. A roll-back operation can be commanded for a patch in either the *Applied* or the *Partial-Apply* states. As the patch is removed, it goes through the following state transitions:

Applied* or *Partial-Apply* to *Partial-Remove

A patch in the *Partial-Remove* state indicates that it has been removed from zero or more, but not from all, the applicable hosts.

Use the command `sw-patch remove` to trigger this transition.

Partial-Remove* to *Available

Use the command `sw-patch host-install` repeatedly targeting each time one of the applicable hosts in the cluster. The transition to the *Available* state is complete when the patch is removed from all target hosts. The patch remains in the patch storage area as if it had just been uploaded.

The following example is basically a copy of the workflow used in *Installing Patches* on page 196, but this time removing instead of installing the patch. The steps are therefore presented in simplified form.

1. Log in as Keystone user **admin** to the active controller.
2. Verify the state of the patch.

```
~(keystone_admin)$ sudo sw-patch query
       Patch ID             Patch State
========================    ===========
PATCH_0001   Applied
```

   In this example the patch is listed in the *Applied* state, but it could be in the *Partial-Apply* state as well.

3. Remove the patch.

```
~(keystone_admin)$ sudo sw-patch remove PATCH_0001
PATCH_0001 has been removed from the repo
```

   The patch is now in the *Partial-Remove* state, ready to be removed from the impacted hosts where it was already installed.

4. Query the patching status of all hosts in the cluster.

```
~(keystone_admin)$ sudo sw-patch query-hosts
  Hostname      IP Address    Patch Current  Reboot Required
============  ==============  =============  ===============
compute-0     192.168.204.15       Yes             No
controller-0  192.168.204.3        No              Yes
controller-1  192.168.204.4        No              Yes
```

In this example, the controllers have patches ready to be removed, and therefore must be rebooted.

5. Remove all pending-for-removal patches from **controller-0**.
   a) Swact controller services.
   b) Run the patching sequence.

```
~(keystone_admin)$ system host-patch-reboot controller-0
```

6. Remove all pending patches from **controller-1**.

The cluster is up to date now. All patches have been removed, and the patch PATCH_0001 can be deleted from the patch storage area if necessary.

It is important to note the role of the command `wrs-path host-install` as a tool that both installs and removes patches as necessary.

Patching Using the Web Administration Interface

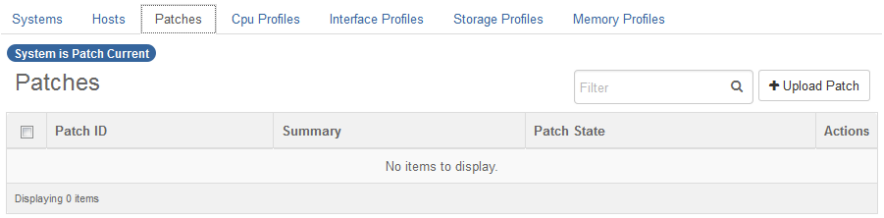
Use the web management interface to upload, delete, apply, and remove patches.

This section presents a example patching workflow using the single patch PATCH_0001.

1. Log in to the web administration interface as the **admin** user.
2. Visit the **Patches** page to see the current patch status.

   Click the **Admin** tab on the System Panel, select **Inventory** to display the **System Inventory** page, and then click the **Patches** tab.

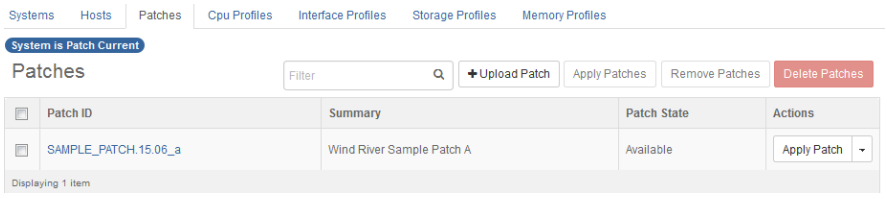
   The **Patches** page is displayed presenting the current status of all patches uploaded to the system. If there are no patches, an empty page is displayed as illustrated below.



3. Upload the patch file to the patch storage area.

   Click the **Upload Patch** button to display an upload window from which you can browse your workstation's file system to select the patch file. Press the **Upload Patch** button once the selection is done.

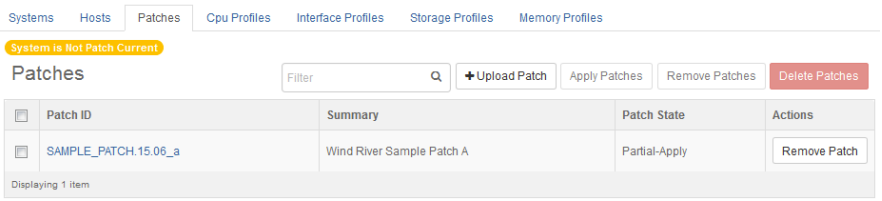
   The patch file is available now in the patch storage area, but it has yet to be applied to the cluster. This is reflected in the **Patches** page as illustrated below.



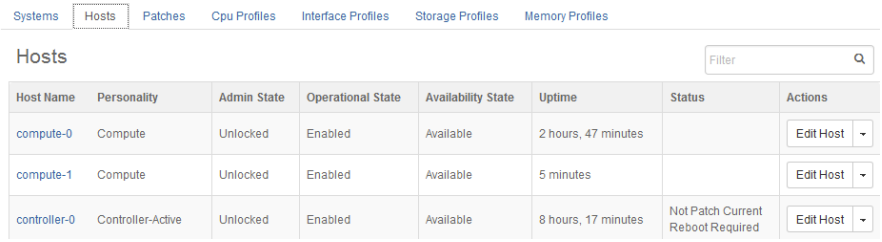
4. Apply the patch.

   Click the **Apply Patch** button associated with the patch. Alternatively, select the patch first using the selection boxes on the left, and then click the **Apply Patches** button at the top. You can use this selection process to apply all patches, or a selected subset, in a single operation.

   The **Patches** page is updated to report the patch to be in the *Partial-Apply* state.



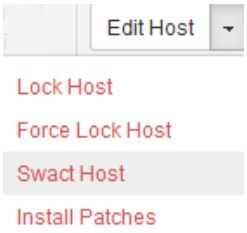
The **Hosts** tab on the **System Inventory** page is also updated to reflect the new status of the hosts with respect to the new patch state. As illustrated in the following image, both controllers are now reported as not patch current and requiring reboot.



5. Install the patch on **controller-0**.

   a) Transfer active services to the standby controller.

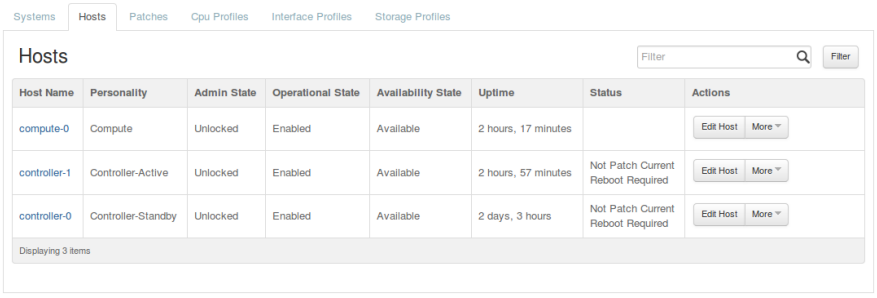
   Swact **controller-0** using the Swact Host option from the **More** button associated with the host.



   Note:

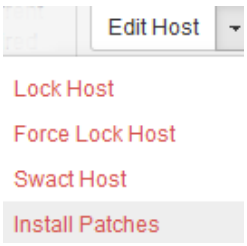
   Access to the web administration interface may be lost briefly during the active controller transition. You may have to log in as user **admin** again.

   After a suitable delay, **controller-1** becomes the active controller and **controller-0** is the standby controller. It is safe then to install the patch.

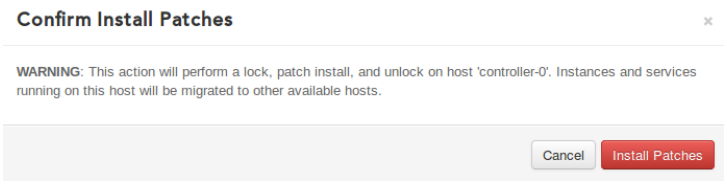


   b) Install the patch.

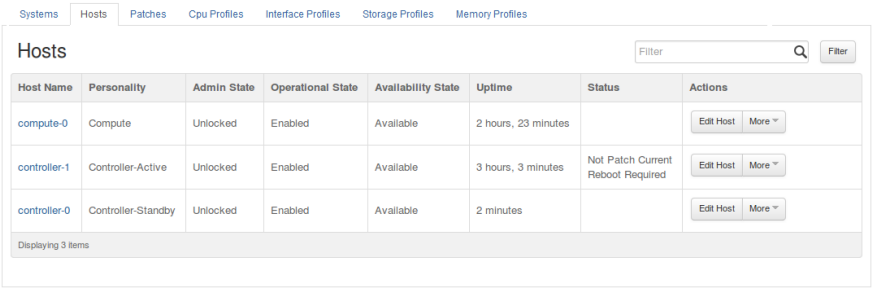
   Select the Install Patches option from the **More** button associated with **controller-0**.



A confirmation window is presented giving you a last opportunity to cancel the operation before proceeding. Note that the patching sequence reboots the host in order to install the patches.



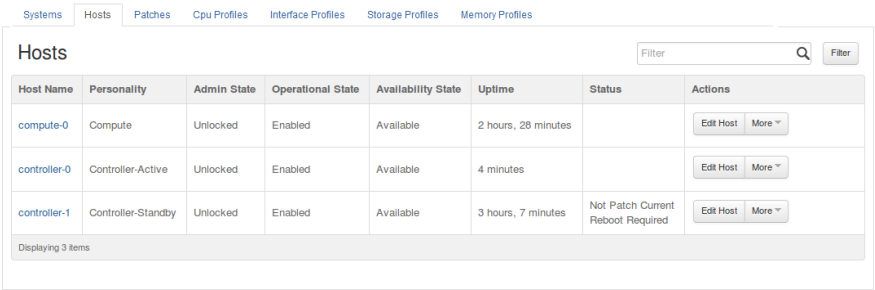
The patch is now installed on **controller-0** which resumes operation as the standby controller.



c) Let **controller-0** resume the active controller role.

Select the Swact Host option again on either controller to transition **controller-0** to the active controller role.

The updated host list looks then as follows:

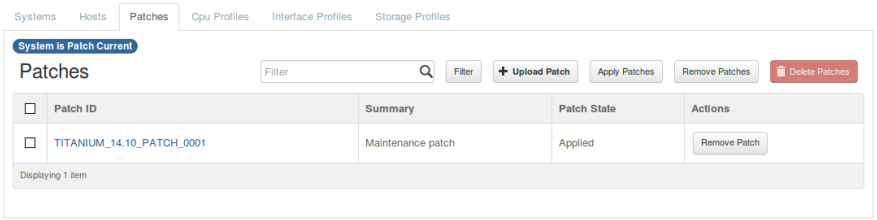


6. Install the patch on **controller-1**.

   Select the Install Patches option from the **More** button associated with **controller-1** in the **Hosts** window, and wait for the host to resume its standby controller role.

7. Verify the state of the patch.

   Visit the **Patches** page again. The patch is now in the *Applied* state as illustrated below.



The patch is applied now, all affected hosts have been updated.

Patches can be removed using the **Remove Patches** button from the **Patches** page. The workflow is similar to the one presented in this section, with the exception that patches are being removed from each host instead of being applied.

Chapter 8

System Backups

Topics:

HP Helion OpenStack Carrier Grade provides tools to back up *system data* and *Cinder volumes* so that they can be stored in external storage resources and used to restore a full cluster.

The backup and restore procedures can be summarized as follows:

- Log in as user **wrsroot** to the console of the active controller.
- Execute the backup procedures from the CLI. The backup files are stored in the directory `/opt/backups` on the current controller node.
- Copy the backup files to an external storage resource.
- Shut down the cluster, and proceed with any maintenance operations that might be required. Ensure that all hosts are powered down.
- Restore the cluster one host at a time. Start with the first controller, then the second controller, then storage nodes, and finally compute nodes.

It is important to note that a restore operation must be executed on the same cluster hardware where the backups were created, otherwise, the operation fails. Hardware differences affect the restore operation in multiple ways. For example, differences in Ethernet MAC addresses impact the networking metadata stored in the system data backup files. Also, storage devices with different sizes or geometries impact the metadata stored in the Cinder block storage database.

The backup and restore tools are designed to operate on an entire cluster. They are not suitable as a tool to back up and restore individual hosts.

Performing a System Data Backup

A system data backup captures core system information needed to restore a fully operational HP Helion OpenStack Carrier Grade cluster.

System data backups include:

- platform configuration details
- system databases
- patching and package repositories
- home directory for the **wrsroot** user and all LDAP user accounts. See *Linux User Accounts* on page 18 for additional information.

System data backups do not include:

- Modifications manually made to the file systems, such as configuration changes on the `/etc` directory. After a restore operation has been completed, these modification have to be reapplied.
- Home directories and passwords of local user accounts. They must be backed up manually by the system administrator.
- The `/root` directory. Use the **wrsroot** account instead when root access is needed.

1. Log in as user **wrsroot** to the active controller.

   You can log in directly on the console or remotely using **ssh**.

2. Execute the system backup.

```
$ sudo config_controller --backup backup_20140918
Performing backup (this will take several minutes):
Step 13 of 13 [###########################################] [100%]
System backup file created: /opt/backups/backup_20140918_system.tgz
Images backup file created: /opt/backups/backup_20140918_images.tgz

Backup complete
```

   In this example, the argument *backup_20140918* to the --backup option is an arbitrary identifier you want to use for the backup.

   Upon successful execution of the command, the following two files are available on the controller's file system:

   - `/opt/backups/backup_20140918_system.tgz`
   - `/opt/backups/backup_20140918_images.tgz`

   Together, these two files represent the entire system data.

3. Transfer the backup files to an external storage resource.

   You can use a command such as `scp` to transfer the backup files to a server reachable over the OAM network. You can also copy them to a locally attached storage device, such as an external USB drive.

4. Optional: Delete the backup files.

The system data backup is now complete. The backup files must be kept in a secured location, probably holding multiple copies of them for redundancy purposes.

Performing a System Restore

You can restore a HP Helion OpenStack Carrier Grade cluster from available system data to bring it back to the operational state it was when the backup procedure took place.

Note:

The following procedure is for use with a system that has separate controller and compute nodes. For a system with combined controller and compute nodes, see *Performing a System Restore for a Minimal Two-server Configuration* on page 212.

Restoring a HP Helion OpenStack Carrier Grade cluster from a set of backup files is done by restoring one host at a time, starting with the controllers, then the storage nodes, and finally the compute nodes.

Before you start the restore procedure you must ensure the following conditions are in place:

- All cluster hosts must be powered down, just as if they were going to be initialized anew.
- All backup files are accessible from a USB flash drive locally attached to the controller where the restore operation takes place (**controller-0**).
- You have the HP Helion OpenStack Carrier Grade installation image available on a USB flash drive, just as when the software was installed the first time. It is mandatory that you use the exact same version of the software used during the original installation, otherwise the restore procedure will fail.
- The restore procedure requires all hosts but **controller-0** to boot over the internal management network using the PXE protocol, just as it was done during the initial software installation. Ideally, you cleaned up all hard drives in the cluster, and the old boot images are no longer present; the hosts then default to boot from the network when powered on. If this is not the case, you must configure each host manually for network booting right after this exercise asks you to power them on.

1. Install the HP Helion OpenStack Carrier Grade software on **controller-0** from the USB flash drive.

   Refer to the *HP Helion OpenStack Carrier Grade Software Installation Guide* for details.

   When the software installation is complete, you should be able to log in using the host's console and the web administration interface.
2. Log in to the console as user **wrsroot** with password **wrsroot**.
3. Ensure the backup files are available to the controller.

   Plug the USB flash drive containing the backup files into the system. The USB flash drive is mounted automatically. Use the command `df` to list the mount points.

   The following steps assume that the backup files are available from a USB flash drive mounted in `/media/wrsroot` in the directory `backups`.
4. Update the controller's software to the previous patching level.

   When restoring the system configuration, the current software version on the controller, at this time, the original software version from the ISO image, is compared against the version available in the backup files. They differ if the latter includes patches applied to the controller's software at the time the backup files were created. If different, the restore process automatically applies the patches and forces an additional reboot of the controller to make them effective.

   The following is the command output if patching is necessary:

```
$ sudo config_controller --restore-system \
/media/wrsroot/backups/titanium_backup_20140918_system.tgz
Restoring system (this will take several minutes):
Step  4 of 19 [#########                                  ] [21%]
This controller has been patched. A reboot is required.
After the reboot is complete, re-execute the restore command.
Enter 'reboot' to reboot controller:
```

   You must enter **reboot** at the prompt as requested. Once the controller is back, log in as user **wrsroot** as before to continue.

   After the reboot, you can verify that the patches were applied, as illustrated in the following example:

```
$ sudo sw-patch query
        Patch ID          Repo State  Patch State
========================  ==========  ===========
COMPUTECONFIG             Available       n/a
```

```
LIBCUNIT_CONTROLLER_ONLY   Applied        n/a
```

5. Restore the system configuration.

   The controller's software is up to date now, at the same stage it was when the backup operation was executed. The restore procedure can be invoked again, and should run without interruptions.

```
$ sudo config_controller --restore-system \
/media/wrsroot/backups/titanium_backup_20140918_system.tgz
Restoring system (this will take several minutes):
Step 19 of 19 [##########################] [100%]
Restoring node states (this will take several minutes):

Locking nodes:
Powering-off nodes:
Step 0 of 0 [##############################################] [100%]

System restore complete
```

6. Authenticate to the system as Keystone user **admin**.

   Source the **admin** user environment as follows:

```
$ cd; source /etc/nova/openrc
```

7. Restore the system images.

```
~(keystone_admin)$ sudo config_controller --restore-images \
/media/wrsroot/backups/titanium_backup_20140918_images.tgz
Step 2 of 2 [##############################################] [100%]

Images restore complete
```

   This step assumes that the backup file resides on the attached USB drive. If instead you copied the image backup file to the `/opt/backups` directory, for example using the `scp` command, you can remove it now.

8. Restore **controller-1**.

   a) List the current state of the hosts.

```
~(keystone_admin)$ system host-list
+----+--------------+-------------+----------------+-------------
+--------------+
| id | hostname     | personality | administrative | operational |
 availability |
+----+--------------+-------------+----------------+-------------
+--------------+
| 1  | controller-0 | controller  | unlocked       | enabled     |
 available    |
| 2  | controller-1 | controller  | locked         | disabled    |
 offline      |
| 3  | compute-0    | compute     | locked         | disabled    |
 offline      |
+----+--------------+-------------+----------------+-------------
+--------------+
```

   b) Power on the host.

      Remember to ensure that the host boots from the network and not from an old disk image that might still be present on its hard drive.

   c) Unlock the host.

```
~(keystone_admin)$ system host-unlock 2
+-----------------+--------------------------------------+
```

```
| Property        | Value                                  |
+-----------------+----------------------------------------+
| action          | none                                   |
| administrative  | locked                                 |
| availability    | online                                 |
| ...             | ...                                    |
| uuid            | 5fc4904a-d7f0-42f0-991d-0c00b4b74ed0 |
+-----------------+----------------------------------------+
```

d) Verify the new state of the hosts.

```
~(keystone_admin)$ system host-list
+----+--------------+-------------+----------------+-------------
+--------------+
| id | hostname     | personality | administrative | operational |
 availability |
+----+--------------+-------------+----------------+-------------
+--------------+
| 1  | controller-0 | controller  | unlocked       | enabled     |
 available    |
| 2  | controller-1 | controller  | unlocked       | enabled     |
 available    |
| 3  | compute-0    | compute     | locked         | disabled    |
 offline      |
+----+--------------+-------------+----------------+-------------
+--------------+
```

The unlocking operation forces a reboot of the host, which is then initialized with the corresponding image available from the system backup.

You must wait for the host to become enabled and available before proceeding to the next step.

9. Restore Cinder volumes.

You restore Cinder volumes by importing them from the backup files. You must import all volume backup files, one at a time.

a) Delete any snapshots that may exist in the system.

The restore operation fails if a snapshot exists for a volume that is about to be restored. You can list the available snapshots with the command `cinder snapshot-list --all-tenants` , and remove them with the command `cinder snapshot-delete` *snapshot-id*.

b) List the current state of the Cinder volumes.

```
~(keystone_admin)$ cinder list --all-tenants
+--------------+----------+--------------+... +----------
+--------------+
|     ID       |   Status  | Display Name |... | Bootable | Attached to
 |
+--------------+----------+--------------+... +----------
+--------------+
| 53ad007a-... |   error   |   volume-1   |... |  false   | c820df55-...
 |
| 578da734-... |   error   |              |... |  true    | c820df55-...
 |
| 593111d8-... |   error   |   volume-2   |... |  true    |
 |
+--------------+----------+--------------+... +----------
+--------------+
```

All volumes are reported to be in the *error* state because they are not available yet, but they are registered in the storage database.

c) Copy the Cinder volumes to the `/opt/backups` folder.

```
~(keystone_admin)$ sudo cp /media/wrsroot/backups/volume-* /opt/backups
```

d) Change to the backups directory, and then import a volume.

```
~(keystone_admin)$ cd /opt/backups
~(keystone_admin)$ cinder import volume-53ad007a-841a-4fd3-
a22b-813d4a6a8a85.tgz
+---------------------+--------------------------------------+
|       Property      |                Value                 |
+---------------------+--------------------------------------+
| display_description |                 None                 |
|          id         | 53ad007a-841a-4fd3-a22b-813d4a6a8a85 |
|         size        |                  1                   |
|        status       |              importing               |
|      updated_at     |      2014-09-19T16:10:24.791220      |
|     volume_type     |                 None                 |
+---------------------+--------------------------------------+
```

The volume remains in the *importing* state while the operation takes place. Use the **backup_status** field on the output of the `cinder show` command to monitor the progress. You must wait until the original volume's state, *in-use* or *available*, is restored. Note that *in-use* volumes are fully restored even if their corresponding instances are not running yet.

e) Import all other volumes in the backup files.

Once finished, all volumes are listed in their original states, as illustrated below.

```
~(keystone_admin)$ cinder list --all-tenants
+--------------+-----------+--------------+... +----------
+--------------+
|      ID      |   Status  | Display Name |... | Bootable | Attached to
 |
+--------------+-----------+--------------+... +----------
+--------------+
| 53ad007a-... |   in-use  |   volume-1   |... |  false   | c820df55-...
 |
| 578da734-... |   in-use  |              |... |  true    | c820df55-...
 |
| 593111d8-... | available |   volume-2   |... |  true    |
 |
+--------------+-----------+--------------+... +----------
+--------------+
```

10. Remove any *in-use* volumes that remain in error.

You must remove all *in-use* volumes that for any reason failed to recover, and their associated virtual machine instances. This is necessary to prevent errors from occurring when restoring the compute nodes used to launch the virtual machines. If an *in-use* volume is in error at the time its virtual machine is launched, the Nova scheduler reports an error and the restore operation fails.

For the purposes of this example, assume that volume **53ad007a-...** failed to restore. Its status is then reported as **error**.

a) Find the associated virtual machine instance.

The ID of the associated instance is available from the output of the `cinder list --all-tenants` command in the previous step. In this case it is **c820df55-...**.

b) Remove the virtual machine instance.

```
~(keystone_admin)$ nova delete c820df55-...
```

c) Remove the volume in error status.

```
~(keystone_admin)$ cinder delete 53ad007a-...
```

11. Restore the compute nodes, one at a time.

You restore these hosts following the same procedure used to restore **controller-1**.

The state of the hosts when the restore operation is complete is as follows:

```
~(keystone_admin)$ system host-list
+----+--------------+-------------+----------------+-------------
+--------------+
| id | hostname     | personality | administrative | operational |
 availability |
+----+--------------+-------------+----------------+-------------
+--------------+
| 1  | controller-0 | controller  | unlocked       | enabled     |
 available    |
| 2  | controller-1 | controller  | unlocked       | enabled     |
 available    |
| 3  | compute-0    | compute     | unlocked       | enabledd    |
 available    |
+----+--------------+-------------+----------------+-------------
+--------------+
```

As each compute node is restored, the original instances at the time the backups were done are started automatically.

The HP Helion OpenStack Carrier Grade is fully restored. The state of the system, including storage resources and virtual machines, is identical to the state the cluster was in when the backup procedure took place.

Passwords for local user accounts must be restored manually since they are not included as part of the backup and restore procedures. See *Linux User Accounts* on page 18 for additional information.

Performing a System Restore for a Minimal Two-server Configuration

You can restore a minimal two-server HP Helion OpenStack Carrier Grade cluster from available system data.

The restore procedure for a minimal-two server configuration differs from the procedure for separate controller and compute nodes. In a system with separate nodes, the controller node is restored first, and then any storage nodes. On systems that use a 3PAR storage backend, these are required to provide the infrastructure used for Cinder volumes. The Cinder volumes are restored next to provide data for VMs. Finally, the compute nodes and any VMs running on them are restored.

For a minimal two-server configuration, the Cinder volumes are restored first to ensure that the VMs on the compute portion of the combined node have access to Cinder volume data. After the Cinder volumes are restored, the combined controller-compute node can be restored.

Before you start the restore procedure you must ensure the following conditions are in place:

- All cluster hosts must be powered down, just as if they were going to be initialized anew.
- All backup files are accessible from a USB flash drive locally attached to the controller where the restore operation takes place (**controller-0**).
- You have the HP Helion OpenStack Carrier Grade installation image available on a USB flash drive, just as when the software was installed the first time. It is mandatory that you use the exact same version of the software used during the original installation, otherwise the restore procedure will fail.
- The restore procedure requires all hosts but **controller-0** to boot over the internal management network using the PXE protocol, just as it was done during the initial software installation. Ideally, you cleaned up all hard drives in the cluster, and the old boot images are no longer present; the hosts then default to boot from the network when

powered on. If this is not the case, you must configure each host manually for network booting right after this exercise asks you to power them on.

1. Install the HP Helion OpenStack Carrier Grade software on **controller-0** from the USB flash drive.

   Refer to the *HP Helion OpenStack Carrier Grade Software Installation Guide* for details.

   When the software installation is complete, you should be able to log in using the host's console and the web administration interface.
2. Log in to the console as user **wrsroot** with password **wrsroot**.
3. Ensure the backup files are available to the controller.

   Plug the USB flash drive containing the backup files into the system. The USB flash drive is mounted automatically. Use the command `df` to list the mount points.

   The following steps assume that the backup files are available from a USB flash drive mounted in `/media/wrsroot` in the directory `backups`.
4. Update the controller's software to the previous patching level.

   When restoring the system configuration, the current software version on the controller, at this time, the original software version from the ISO image, is compared against the version available in the backup files. They differ if the latter includes patches applied to the controller's software at the time the backup files were created. If different, the restore process automatically applies the patches and forces an additional reboot of the controller to make them effective.

   The following is the command output if patching is necessary:

   ```
   $ sudo config_controller --restore-system \
   /media/wrsroot/backups/titanium_backup_20140918_system.tgz
   Restoring system (this will take several minutes):
   Step  4 of 19 [#########                                   ] [21%]
   This controller has been patched. A reboot is required.
   After the reboot is complete, re-execute the restore command.
   Enter 'reboot' to reboot controller:
   ```

   You must enter **reboot** at the prompt as requested. Once the controller is back, log in as user **wrsroot** as before to continue.

   After the reboot, you can verify that the patches were applied, as illustrated in the following example:

   ```
   $ sudo sw-patch query
           Patch ID          Repo State  Patch State
   ========================  ==========  ===========
   COMPUTECONFIG             Available       n/a
   LIBCUNIT_CONTROLLER_ONLY   Applied        n/a
   ```
5. Restore the system configuration.

   The controller's software is up to date now, at the same stage it was when the backup operation was executed. The restore procedure can be invoked again, and should run without interruptions.

   ```
   $ sudo config_controller --restore-system \
   /media/wrsroot/backups/titanium_backup_20140918_system.tgz
   Restoring system (this will take several minutes):
   Step 19 of 19 [##########################] [100%]
   Restoring node states (this will take several minutes):

   Locking nodes:
   Powering-off nodes:
   Step 0 of 0 [##############################################] [100%]

   System restore complete
   ```

6. Authenticate to the system as Keystone user **admin**.

   Source the **admin** user environment as follows:

   ```
   $ cd; source /etc/nova/openrc
   ```

7. Restore the system images.

   ```
   ~(keystone_admin)$ sudo config_controller --restore-images \
   /media/wrsroot/backups/titanium_backup_20140918_images.tgz
   Step 2 of 2 [###############################################] [100%]

   Images restore complete
   ```

   This step assumes that the backup file resides on the attached USB drive. If instead you copied the image backup file to the `/opt/backups` directory, for example using the `scp` command, you can remove it now.

8. Restore Cinder volumes.

   You restore Cinder volumes by importing them from the backup files. You must import all volume backup files, one at a time.

   a) Delete any snapshots that may exist in the system.

      The restore operation fails if a snapshot exists for a volume that is about to be restored. You can list the available snapshots with the command `cinder snapshot-list --all-tenants` , and remove them with the command `cinder snapshot-delete` *snapshot-id*.

   b) List the current state of the Cinder volumes.

   ```
   ~(keystone_admin)$ cinder list --all-tenants
   +--------------+-----------+--------------+... +----------
   +--------------+
   |     ID       |   Status  | Display Name |... | Bootable | Attached to
    |
   +--------------+-----------+--------------+... +----------
   +--------------+
   | 53ad007a-... |   error   |   volume-1   |... |  false   | c820df55-...
    |
   | 578da734-... |   error   |              |... |  true    | c820df55-...
    |
   | 593111d8-... |   error   |   volume-2   |... |  true    |
    |
   +--------------+-----------+--------------+... +----------
   +--------------+
   ```

   All volumes are reported to be in the *error* state because they are not available yet, but they are registered in the storage database.

   c) Copy the Cinder volumes to the `/opt/backups` folder.

   ```
   ~(keystone_admin)$ sudo cp /media/wrsroot/backups/volume-* /opt/backups
   ```

   d) Change to the backups directory, and then import a volume.

   ```
   ~(keystone_admin)$ cd /opt/backups
   ~(keystone_admin)$ cinder import volume-53ad007a-841a-4fd3-
   a22b-813d4a6a8a85.tgz
   +---------------------+--------------------------------------+
   |       Property      |                Value                 |
   +---------------------+--------------------------------------+
   | display_description |                 None                 |
   |          id         | 53ad007a-841a-4fd3-a22b-813d4a6a8a85 |
   |         size        |                  1                   |
   |        status       |              importing               |
   ```

```
|      updated_at     |      2014-09-19T16:10:24.791220      |
|     volume_type     |                 None                 |
+---------------------+--------------------------------------+
```

The volume remains in the *importing* state while the operation takes place. Use the **backup_status** field on the output of the `cinder show` command to monitor the progress. You must wait until the original volume's state, *in-use* or *available*, is restored. Note that *in-use* volumes are fully restored even if their corresponding instances are not running yet.

e) Import all other volumes in the backup files.

Once finished, all volumes are listed in their original states, as illustrated below.

```
~(keystone_admin)$ cinder list --all-tenants
+--------------+-----------+--------------+... +----------
+--------------+
|      ID      |   Status  | Display Name |... | Bootable | Attached to
 |
+--------------+-----------+--------------+... +----------
+--------------+
| 53ad007a-... |   in-use  |   volume-1   |... |  false   | c820df55-...
 |
| 578da734-... |   in-use  |              |... |  true    | c820df55-...
 |
| 593111d8-... | available |   volume-2   |... |  true    |
 |
+--------------+-----------+--------------+... +----------
+--------------+
```

9. Remove any *in-use* volumes that remain in error.

You must remove all *in-use* volumes that for any reason failed to recover, and their associated virtual machine instances. This is necessary to prevent errors from occurring when restoring the compute functions used to launch the virtual machines. If an *in-use* volume is in error at the time its virtual machine is launched, the Nova scheduler reports an error and the restore operation fails.

For the purposes of this example, assume that volume **53ad007a-...** failed to restore. Its status is then reported as **error**.

a) Find the associated virtual machine instance.

The ID of the associated instance is available from the output of the `cinder list --all-tenants` command in the previous step. In this case it is **c820df55-...**.

b) Remove the virtual machine instance.

```
~(keystone_admin)$ nova delete c820df55-...
```

c) Remove the volume in error status.

```
~(keystone_admin)$ cinder delete 53ad007a-...
```

10. Restore the compute function on **controller-0**.

Use the following command:

```
~(keystone_admin)$ system compute-config-complete
Manifests are being created and applied, the controller will restart upon
    completion.
+-------------------+--------------------------------------+
| Property          | Value                                |
+-------------------+--------------------------------------+
| action            | none                                 |
| administrative    | unlocked                             |
| availability      | available                            |
```

```
| bm_ip              |                                      |
| bm_mac             | None                                 |
| bm_type            | None                                 |
| bm_username        | None                                 |
| capabilities       | {}                                   |
| created_at         | 2015-07-07T20:47:40.329544+00:00     |
| cstatus            | Compute config required              |
| hostname           | controller-0                         |
| iconfig_applied    | 4e8acad4-0d68-490b-afb3-33dc3fbac3c1 |
| iconfig_fini       | None                                 |
| iconfig_target     | None                                 |
| id                 | 1                                    |
| infra_ip           | None                                 |
| invprovision       | provisioned                          |
| location           | {}                                   |
| mgmt_ip            | 192.168.204.3                        |
| mgmt_mac           | 08:00:27:58:27:b8                    |
| operational        | enabled                              |
| personality        | controller                           |
| reserved           | False                                |
| serialid           | None                                 |
| subfunction_avail  | not-installed                        |
| subfunction_oper   | disabled                             |
| subfunctions       | controller,compute                   |
| task               |                                      |
| updated_at         | 2015-07-07T20:55:43.926852+00:00     |
| uptime             | 1370                                 |
| uuid               | bd213301-0130-4698-89eb-8a58386ffe03 |
+--------------------+--------------------------------------+
```

11. Restore **controller-1**.

a) List the current state of the hosts.

```
~(keystone_admin)$ system host-list
+----+--------------+-------------+----------------+-------------
+--------------+
| id | hostname     | personality | administrative | operational |
 availability |
+----+--------------+-------------+----------------+-------------
+--------------+
| 1  | controller-0 | controller  | unlocked       | enabled     |
 available    |
| 2  | controller-1 | controller  | locked         | disabled    |
 offline      |
+----+--------------+-------------+----------------+-------------
+--------------+
```

b) Power on the host.

Remember to ensure that the host boots from the network and not from an old disk image that might still be present on its hard drive.

c) Unlock the host.

```
~(keystone_admin)$ system host-unlock 2
+------------------+--------------------------------------+
| Property         | Value                                |
+------------------+--------------------------------------+
| action           | none                                 |
| administrative   | locked                               |
| availability     | online                               |
| ...              | ...                                  |
| uuid             | 5fc4904a-d7f0-42f0-991d-0c00b4b74ed0 |
```

```
+-----------------+--------------------------------------+
```

d) Verify the new state of the hosts.

```
~(keystone_admin)$ system host-list
+----+--------------+-------------+----------------+-------------
+--------------+
| id | hostname     | personality | administrative | operational |
 availability |
+----+--------------+-------------+----------------+-------------
+--------------+
| 1  | controller-0 | controller  | unlocked       | enabled     |
 available    |
| 2  | controller-1 | controller  | unlocked       | enabled     |
 available    |
+----+--------------+-------------+----------------+-------------
+--------------+
```

The unlocking operation forces a reboot of the host, which is then initialized with the corresponding image available from the system backup.

You must wait for the host to become enabled and available before proceeding to the next step.

The HP Helion OpenStack Carrier Grade is fully restored. The state of the system, including storage resources and virtual machines, is identical to the state the cluster was in when the backup procedure took place.

Passwords for local user accounts must be restored manually since they are not included as part of the backup and restore procedures. See *Linux User Accounts* on page 18 for additional information.

Chapter 9

Managing Stacks

Topics:

You can create and manage collections of resources, or *stacks*, using Heat, the OpenStack orchestration service. HP Helion OpenStack Carrier Grade extensions are included for enhanced scope and reliability.

With Heat, you can define a stack configuration in a *template* file, and then apply the template to create or modify the stack resources and connections. The Heat orchestration layer includes life-cycle management features to simplify the addition, modification, and deletion of stacks.

Stacks

A collection of resources created and managed using the Heat orchestration service is called a *stack*.

Stack resources can be flavors, images, instances, tenant networks, volumes, security groups, users, floating IP addresses, or any other entities defined and configured in Heat templates. Most types of resources offered by OpenStack can be defined in Heat templates and included in stacks.

Using the CLI, you can obtain information on existing stacks in several ways:

- `heat stack-list` shows the names of stacks
- `heat stack-show` *name* shows details about the named stack
- `heat stack-create` *name* creates the named stack with given parameters
- `heat stack-modify` *name* modifies the named stack according to given parameters
- `heat stack-delete` *name* deletes the named stack

Stacks can include *static* and *autoscaling* resources.

Static resource

A resource that is defined when the stack is launched, and does not change unless a `heat stack-modify` command is issued.

Autoscaling resource

A resource that can be created, modified, or removed in response to performance metrics collected by Ceilometer. These can include metrics generated by the HP Helion OpenStack Carrier Grade or OpenStack platform, and metrics generated by VM instances using CloudWatch.

Templates

A template is a formal set of instructions for defining and configuring a stack.

Templates specify the resources to include in a stack, and relationships between them.

A template contains several sections, including:

Description

You can use this section to provide comments about the stack, such as its purpose or any special considerations when using it.

Parameters

This section defines the parameters that you can pass to the stack template. You can pass parameters using the command line, a web form, or an environment file.

You can include a default value for a parameter, and optionally pass a different value when the template is deployed. If no default value is included, you must provide a value when the template is deployed. Parameters without default values are called *mandatory parameters*.

Resources

This section defines the resources to be created. They can be specified in any order.

There are two official formats for templates: CloudFormation (CFN) and Heat Orchestration Template (HOT).

CFN

This format is associated with Amazon Web Services (AWS) technology. A CFN template is indicated by the first line:

```
HeatTemplateFormatVersion: '2012-12-12'
```

HOT

This format is associated specifically with the OpenStack project. A HOT template is indicated by the first line:

```
heat_template_version: 2013-05-23
```

In HOT, some of the built-in functions are different, and lowercase is used for attribute names.

Both formats support Amazon Web Services (AWS) resources and OpenStack resources, in any combination.

HP Helion OpenStack Carrier Grade extensions to Heat functionality are supported only on OpenStack resources, which are the resources primarily used with HOT format.

HP Helion OpenStack Carrier Grade includes sample stack templates in HOT format for demonstration purposes (see *Sample Templates for HP Helion OpenStack Carrier Grade* on page 226). Example CFN-formatted Heat templates can be found at *https://github.com/openstack/heat-templates/tree/master/cfn*. The templates are expressed using YAML, which is the recommended notation for HOT. Heat templates may also be expressed using JSON, which is the usual notation for CFN.

For more information on the content and syntax of Heat templates, see the online *OpenStack Template Guide*.

Parameters

You can specify parameters for a stack, such as authentication, storage, networking, and so on, when you invoke a template.

You can supply parameter values by:

- including them in the template as defaults
- specifying each one using a form in the web administration interface, or using the -parameters option on the `heat create-stack` command line
- saving them in an environment file, which you can specify using the -e option on the `heat create-stack` command line

This enables you to customize the behavior of a common stack template when the stack is created or modified.

You can use environment files or command-line entries to supplement parameters in the template, or to override them if they already exist. Environment-file values override existing values in the template. Command-line values override existing values in the environment file or the template.

The syntax for specifying parameters is as follows:

- on the command line, using the -P option. Multiple -P arguments are supported.

```
~(keystone_admin)$ heat stack-create -P key1=val1 -P key2=val2
```

- in an environment file:

```
parameters:
    key:value
    key:value
```

HP Helion OpenStack Carrier Grade Extensions to Heat

Several extensions to Heat are included with HP Helion OpenStack Carrier Grade.

The HP Helion OpenStack Carrier Grade Extensions are supported only for OpenStack resources, which are the primary resources used with HOT format.

Multiple NIC Support

When launching a VM instance initially or in an autoscaling stack, you can specify multiple network interfaces. The open-source version of Heat allows only a single network interface to be specified at launch.

The syntax for the extension is as follows:

```
...
vm:
    type: OS::Nova::Server
    properties:
      ...
      networks :
      - { network: 'tenant1-mgmt-net', vif-model: virtio}
      - { network: { get_param: port0 }, vif-model: avp}
      - { network: { get_param: port1 }, vif-model: avp}
      ...
```

As the example shows, this extension also adds the ability to specify a different vif-model for each interface. In addition, a new avp option is introduced, supporting the use of optimized AVP device drivers.

The valid vif-model values are as follows:

- avp (Accelerated Virtual Port)
- e1000 (Intel e1000 Emulation)
- ne2k_pci (NE2000 Emulation)
- pcnet (AMD PCnet/PCI Emulation)
- rtl8139 (Realtek 8139 Emulation)
- virtio (VirtIO Network)
- pci-passthrough (PCI Passthrough Device)
- pci-sriov (SR-IOV device)

Simplified VM Instance Naming

HP Helion OpenStack Carrier Grade introduces minor changes to the OpenStack VM naming convention to make Heat-generated names more user-friendly.

- For a static resource, launched VM instances are named using the name attribute of the OS::Nova::Server structure (without including the `<StackTemplateName>`).

```
Serving_Gateway:
    Type: OS::Nova::Server
    Properties:
      name  : ‘Serving_Gateway’
      image : 'cirros'
      flavor: 'm1.tiny'


~(keystone_admin)$ heat stack-create –f <file> EPC
~(keystone_admin)$ nova list
+--------------------------------------+-----------------+--------+-…
| ID                                   | Name            | Status | …
+--------------------------------------+-----------------+--------+-…
| 581b3495-3cf1-4410-9587-5cf04fccfed2 | Serving_Gateway | ACTIVE | …
+--------------------------------------+-----------------+--------+-…
```

Enhanced Support for Server Groups

You can specify a maximum group size and a best-effort attribute for Server Groups.

The syntax for adding a Server Group Resource is as follows:

```
...
resources:
    ...
    my_server_group:
        type: OS::Nova::ServerGroup
        properties:
            policy: 'anti-affinity'
            wrs-sg:group_size: 4
            wrs-sg:best_effort: False
    ...
```

Relaxed Requirements for Passing User Data

The property UserDataType is a HP Helion OpenStack Carrier Grade template extension that you can use to pass user data to an instance even if the instance does not have **cloud-init** installed. For more information, see *Customizing Guest Images with User Data* on page 225.

Improved User Access to Stacks

Stacks can be created, modified, or deleted by admin or non-admin users.

Greater Control over Resource Allocations

When creating a network resource using OS::Neutron::Net, you can use a **depends_on** attribute to ensure that the requirements of other resources are given priority before the resource is created. The attribute takes another resource as an argument. In the following example, it is used to specify that the resource **external_network** must be created before **internal_network** is created.

```
internal_network:
      type: OS::Neutron::Net
      properties:
        name: { get_param: INTERNALNET }
        depends_on: { get_resource: external_network }
        shared: false
        tenant_id: {get_param: TENANT_ID}
```

Support for Local Files in Glance Image

Support is added for specifying a local file or web URL as the location of a Glance image.

```
...
wrl5_Glance_Image:
   type: OS::Glance::Image
   properties:
     name: {get_param: IMAGE_NAME }
     is_public: true
     container_format : bare
     disk_format : qcow2
     location: file:///home/wrsroot/images/cgcs-guest.img
...
```

Support for name Property in Nova Flavor

A **name** property is added to the Flavor resource to support user-friendly names for Heat-generated flavors.

Additional Heat Resources

In addition to the standard OpenStack resources available for Heat templates, you can use the following resources:

OS::WR::ScalingPolicy

Defines a Resource UP/Down autoscaling property.

WR::Neutron::ProviderNet

Defines a provider network.

WR::Neutron::ProviderNetRange

Defines a segmentation range for a provider network.

WR::Neutron::QoSPolicy

Defines a packet scheduling weight that can be referenced by a tenant network (OS::Neutron::Net).

WR::Neutron::PortForwarding

Defines a destination NAT mapping between an external IP address (a router address) and external port, and an internal (VM) IP address and port.

Launching a Stack

You can launch stacks using the web administration interface, or you can use the command-line interface.

These instructions assume you are using the web administration interface. For CLI assistance, use the following command.

```
~(keystone_admin)$ heat help stack-create
```

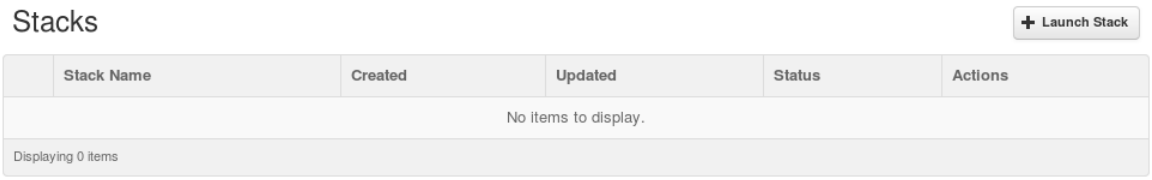
Ensure that you have a template for the stack.

Depending on the template, you may need to collect some parameters in advance. For details, see the documentation for the specific template, or refer to the Parameters section of the template.

1. Open the list of existing stacks.

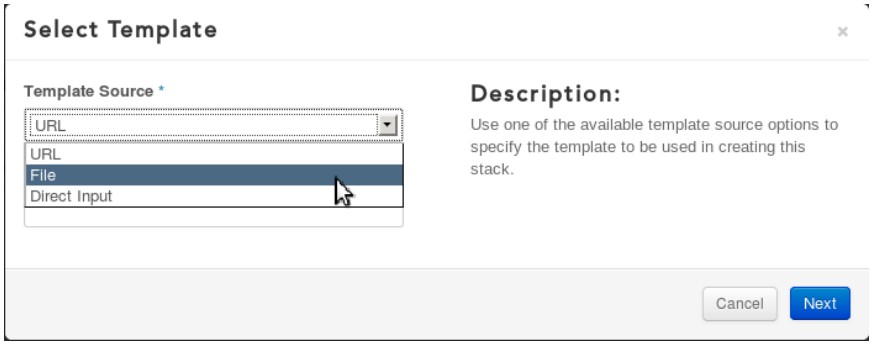
   From the **Project** menu of the web administration interface, select **Orchestration** > **Stacks**.

   The Stacks list appears.



2. Launch a new stack and specify a template.

   Click **Launch Stack**, and then complete the **Select Template** dialog box.



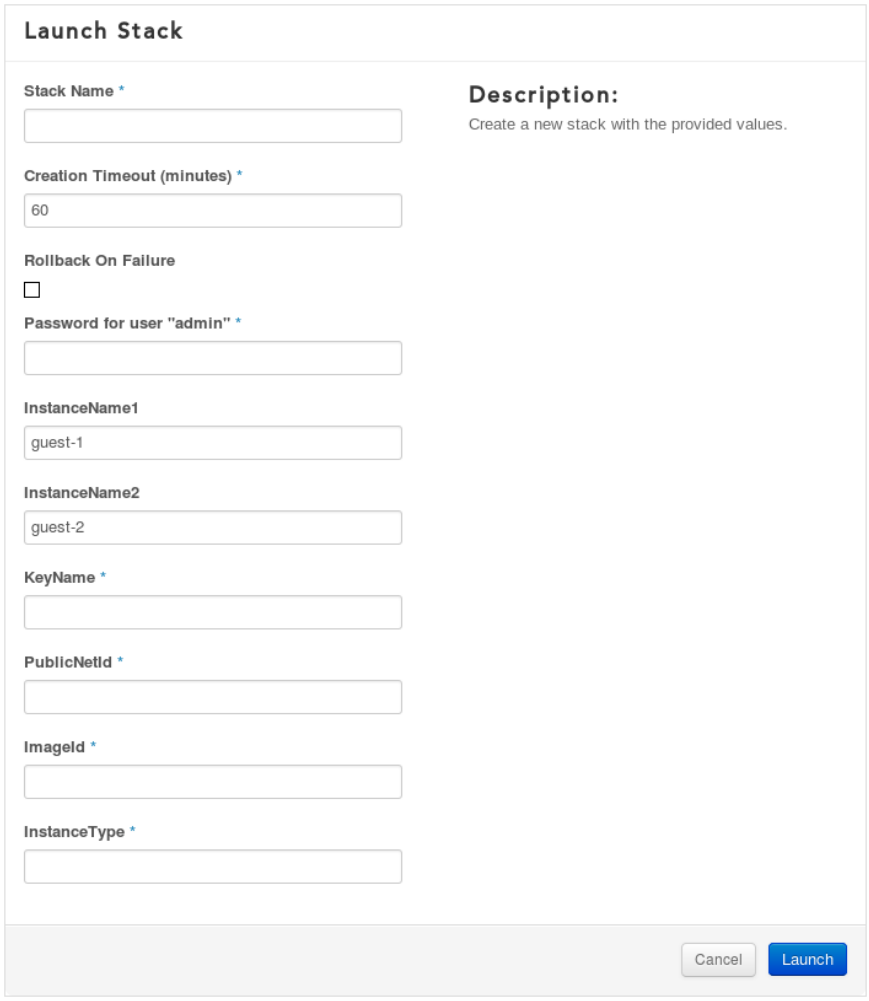
The **Template Source** drop-down list offers several ways to specify a template.

Option	Description
URL	Provides a text field for URL access to a template file.
File	Provides a **Browse** button for access to locally stored template files.
Direct Input	Provides a text window for direct entry of template content.

3. Complete the **Launch Stack** form to provide any parameter values required by the template.

   Note:

   The form is created dynamically based on the Parameters section of the template, and varies depending on the template. The form shown here is an example only.



Customizing Guest Images with User Data

You can provide bootstrap configuration for an instance at launch by including user data in the template.

You can include user data by defining a **UserData** property for an instance. This sends configuration instructions to the instance at launch. For an example of user data, see the `CombinationAutoScaling.yaml` template included with HP Helion OpenStack Carrier Grade.

Normally, this requires **cloud-init** to be installed in the guest, because the user data is converted to MIME format and then interpreted by **cloud-init**. For instances that do not include **cloud-init**, you can bypass the MIME conversion of user data by invoking an extension property and assigning the value **'RAW'**. This allows the VM to retrieve the specified user data in unaltered format through a simple REST API call to the metadata server. The name of the property follows the convention for the associated resource type. For an AWS resource (for example, **AWS::EC2::Instance**), use UserDataType. For an OpenStack resource (for example, **OS::Nova::Server**), use user_data_type. The following code fragment shows the property used with an AWS resource.

```
...
Srv:
   type: OS::Nova::Server
    properties:
      name: l3-forwarding
      …
      user_data_format: 'RAW'
      user_data:
```

```
        Fn::Base64:
            Fn::Replace:
            - 'OS::stack_name': {Ref: 'OS::stack_name'}
            - |
              #cloud-config

              runcmd:
               - ifconfig eth1 1.1.1.1/24 up
               - ifconfig eth2 2.2.2.1/24 up
               - echo 1 > /proc/sys/net/ipv4/ip_forward
               - sed -i -e 's#.*\(net.ipv4.ip_forward\).*#\1 = 1#g' /etc/
sysctl.conf
               - ip route add 192.50.0.0/16 via 1.1.1.10
               - ip route add 192.51.0.0/16 via 2.2.2.10
...
```

Note:

For a VM to access user data, it must have a DHCP-enabled interface on a tenant network that has a Neutron router. Typically, this is the interface to the VM's OAM network or an internal network. The Neutron router provides a route to the metadata server, which provides instances with access to user data.

Resource Scaling (Autoscaling)

You can use Heat to reassign stack resources automatically to meet changing conditions.

You can define and monitor performance thresholds for metrics such as CPU activity, and then add or remove resources when the thresholds are crossed. This allows you to make efficient use of the hardware in the cluster, by allocating resources only when they are needed, and assigning them where they are most required.

HP Helion OpenStack Carrier Grade supports two types of scaling:

In/Out

This type of scaling (also known as *horizontal scaling*) adds or removes instances as needed.

Up/Down

This type of scaling (also known as *vertical scaling*) increases or decreases resources (for example, vCPUs) for individual instances as needed. For more about up/down scaling, see *Scaling Virtual Machine Resources* on page 162.

Performance metrics can be collected and reported by the HP Helion OpenStack Carrier Grade platform, or by the guests using guest metrics.

Sample Templates for HP Helion OpenStack Carrier Grade

You can evaluate selected features of Heat using sample templates. The samples also demonstrate some HP Helion OpenStack Carrier Grade extensions.

Sample Heat templates are included with the HP Helion OpenStack Carrier Grade SDK. A brief description is provided for each sample. For more information about the SDK, refer to the *Guest Software Development Kit* documentation.

The samples are also copied to the HP Helion OpenStack Carrier Grade controllers at system installation. They can be found in the `/etc/heat/templates/hot` directory.

Creating a Static Resource

The `BootFromCinder.yaml` template illustrates the use of templates to create static resources.

This HOT template creates a volume and a VM using this volume. The VM name is specified as a parameter in the template, illustrating the use of HP Helion OpenStack Carrier Grade extensions to simplify VM instance naming. It is used by the **OS::Nova::Server** resources, which define the construction of the VMs.

The VM has two attached networks.

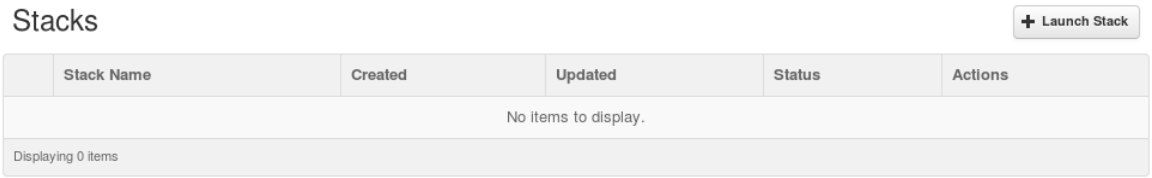
1. Collect the required parameter values for this template.

   For information about flavors, keypairs, images, or tenant networks, see the *HP Helion OpenStack Carrier Grade Reference Deployment Scenarios: User Tasks*.

   a) On the **Admin** > **System Panel** > **Flavors** page, locate a **Flavor Name** (a resource profile).
   b) On the **Project** > **Compute** > **Access & Security** page, locate a **Keypair Name** to provide secure access to the stack.
   c) On the **Admin** > **System Panel** > **Images** page, locate an **Image Name** to specify the VM image used by the stack.
   d) On the **Admin** > **System Panel** > **Networks** page, click the appropriate network **Name** to open the Network Overview page, and then obtain the network **ID**.
   e) On the same page, locate and record an internal network to provide connectivity for the stack.
2. Open the list of existing stacks.

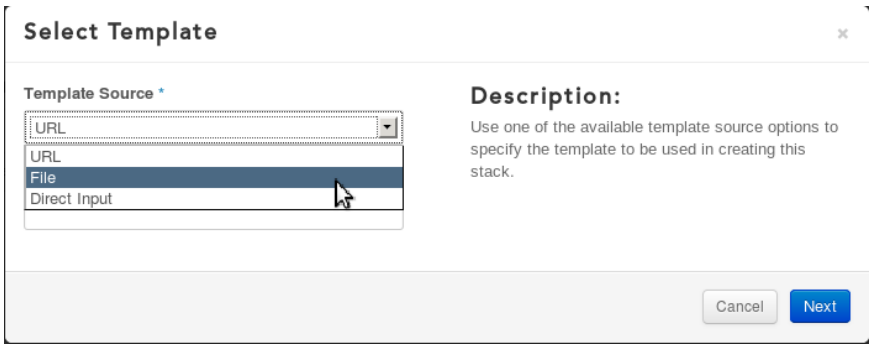
   From the **Project** menu of the web administration interface, select **Orchestration** > **Stacks**.

   The Stacks list appears.



3. Launch a new stack and specify a template.

   Click **Launch Stack**, and then complete the **Select Template** dialog box.

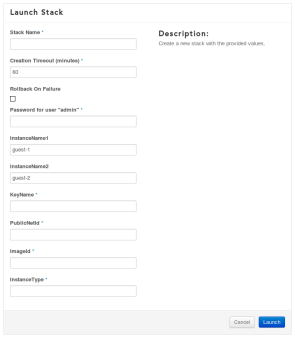


The **Template Source** drop-down list offers several ways to specify a template.

Option	Description
URL	Provides a text field for URL access to a template file.
File	Provides a **Browse** button for access to locally stored template files.

Option	Description
Direct Input	Provides a text window for direct entry of template content.

4. Select the template, then click **Browse** and select `BootFromCinder.yaml` from the local disk.
5. Use the **Launch Stack** form to provide the required parameter values for the template.



Creating an In/Out Autoscaling Resource

The `CombinationAutoscaling.yaml` template illustrates the use of a template to create an in/out autoscaling resource.

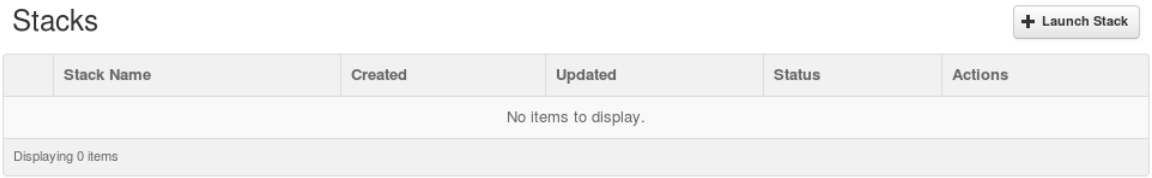
This HOT template creates a load balancer VM (**OS::Nova::Server**) and a scalable pool of server VMs (**OS::Heat::AutoScalingGroup** of **OS::Nova:Server**). The associated OS::Ceilometer:Alarm and OS::Heat::ScalingPolicy resources are based on a passed-in platform link utilization meter. Instances are added or removed based on this information.

Use an **admin** account to run this template.

1. Collect the required parameter values for this template.
   a) On the **Admin** > **System Panel** > **Flavors** page, locate a **Flavor Name** (a resource profile).
   b) On the **Project** > **Compute** > **Access & Security** page, locate a **Keypair Name** to provide secure access to the stack.
   c) On the **Admin** > **System Panel** > **Images** page, locate an **Image Name** to specify the VM image used by the stack.
   d) On the **Admin** > **System Panel** > **Networks** page, click the **Name** of each required network in turn, in order to open the Network Overview page and obtain the network **ID**. For this template, a public network ID and an internal network ID are required.
2. Open the list of existing stacks.

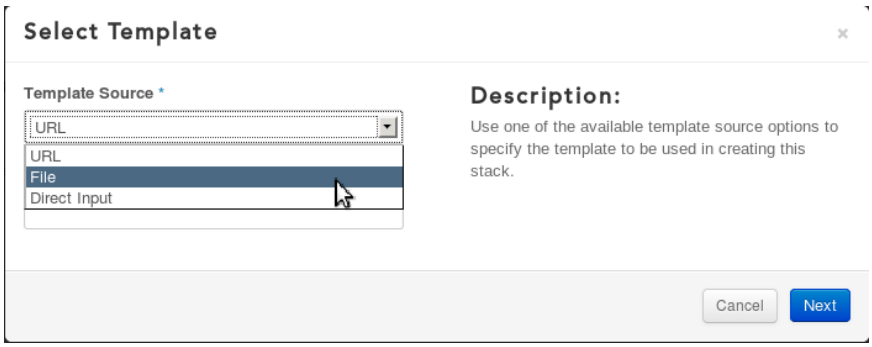
   From the **Project** menu of the web administration interface, select **Orchestration** > **Stacks**.

   The Stacks list appears.



3. Launch a new stack and specify a template.

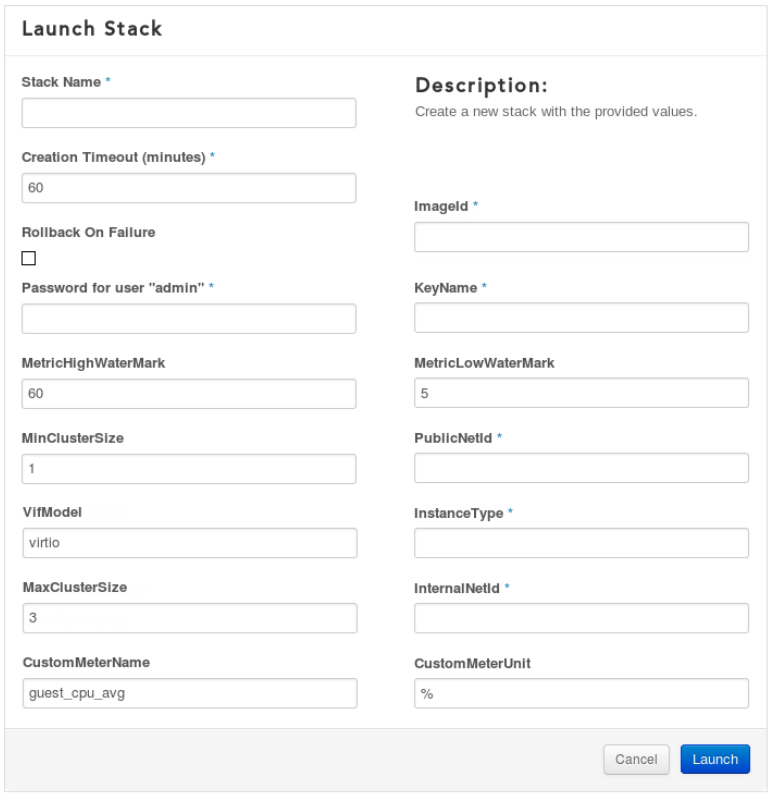
   Click **Launch Stack**, and then complete the **Select Template** dialog box.



The **Template Source** drop-down list offers several ways to specify a template.

Option	Description
URL	Provides a text field for URL access to a template file.
File	Provides a **Browse** button for access to locally stored template files.
Direct Input	Provides a text window for direct entry of template content.

4. Select **File**, then click **Browse** and select `CombinationAutoscaling.yaml` from the local disk.
5. Use the **Launch Stack** form to provide the required parameter values for the template.



Note:

The parameters are presented in a single column. This image has been edited to present them in two columns.

6. Optional: Adjust existing values to meet your requirements.

Option	Description
KeyName	The **KeyPair** name from the **Access & Security** page.
ViFModel	The virtual interface model to use. For valid choices, see *Multiple NIC Support* in *HP Helion OpenStack Carrier Grade Extensions to Heat* on page 220.
InstanceType	The **Flavor Name** from the **Flavors** page.
ImageId	The **Image Name** from the **Images** page.
PublicNetId	The **ID** from the **Network Overview** page.
InternalNetId	The **ID** from the **Network Overview** page.
MinClusterSize	The minimum number of VMs to launch. The stack cannot autoscale below this value.
MaxClusterSize	The maximum number of VMs that can be launched.
CustomMeterName	The name of a custom Ceilometer metric to store.
CustomMeterUnit	The unit of measurement to display for the custom metric; for example, 'Percent'.
MetricHighWaterMark	The value that triggers autoscaling to add resources.
MetricLowWaterMark	The value that triggers autoscaling to remove resources.

Creating an Up/Down Autoscaling Resource

The `VMAutoscaling.yaml` template illustrates the use of a template to create an up/down autoscaling resource.

This HOT template creates a VM instance with vCPU autoscaling. The number of online vCPUS for the instance is scaled up or down depending on vCPU utilization. The scaling range (minimum and maximum number of vCPUs) is determined by the launch flavor.

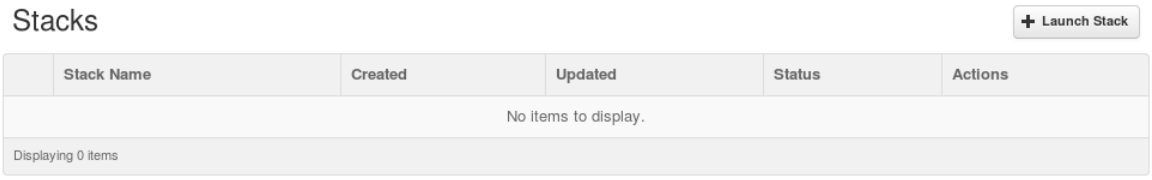
For up/down scaling, the CPU measurement uses a **vcpu_util** meter, which takes into account the current number of online vCPUs when measuring usage. This is in contrast to the **cpu_util** meter, which always uses the maximum number of vCPUs. Using **cpu_util** for up/down scaling would produce an inaccurate usage/availability ratio if any vCPUs were offline, since it assumes all VCPUs are available.

The VM image selected for launch must be capable of scaling. For more information about VM scaling, see *Scaling Virtual Machine Resources* on page 162.

1. Collect the required parameters.
   a) On the **Admin** > **System Panel** > **Flavors** page, locate a **Flavor Name** (a resource profile).
   b) On the **Project** > **Compute** > **Access & Security** page, locate a **Keypair Name** to provide secure access to the stack.
   c) On the **Admin** > **System Panel** > **Images** page, locate an **Image Name** to specify the VM image used by the stack.
   d) On the **Admin** > **System Panel** > **Networks** page, obtain the **Name** of the private network.
   e) On the **Admin** > **Identity Panel** > **Users** page, locate the **User Name** for the tenant where the VM will be launched.
2. Open the list of existing stacks.

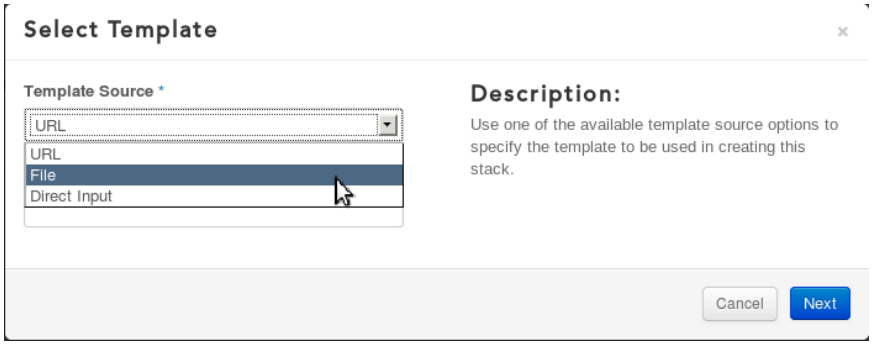
From the **Project** menu of the web administration interface, select **Orchestration** > **Stacks**.

The Stacks list appears.



3. Launch a new stack and specify a template.

   Click **Launch Stack**, and then complete the **Select Template** dialog box.



   The **Template Source** drop-down list offers several ways to specify a template.

Option	Description
URL	Provides a text field for URL access to a template file.
File	Provides a **Browse** button for access to locally stored template files.
Direct Input	Provides a text window for direct entry of template content.

4. Select **File**, then click **Browse** and select `VMAutoScaling.yaml` from the local disk.
5. Adjust the parameters in the **Launch Stack** form to meet your requirements.

Option	Description
KeyName	The **KeyPair** name from the **Access & Security** page.
TenantUserID	The tenant user ID associated with the instance launch. Scaling of the instance resources can only be performed by the tenant associated with the launch.
FlavorName	The **Flavor Name** from the **Flavors** page.
ImageName	The **Image Name** from the **Images** page.
PrivateNetName	The **Name** from the **Network Overview** page.
InstanceName	The name of the instance to create.
CustomMeterName	The name of a custom Ceilometer metric to store.

Option	Description
CustomMeterUnit	The unit of measurement to display for the custom metric; for example, 'Percent'.
MetricHighWaterMark	The value that triggers autoscaling to add resources.
MetricLowWaterMark	The value that triggers autoscaling to remove resources.

Supported Heat Resource Types

The HP Helion OpenStack Carrier Grade implementation of Heat has been tested with most commonly-used resource types.

The following resource types have been tested for use with HP Helion OpenStack Carrier Grade.

- AWS::AutoScaling::AutoScalingGroup
- AWS::AutoScaling::LaunchConfiguration
- AWS::AutoScaling::ScalingPolicy
- AWS::CloudFormation::Stack
- AWS::EC2::Instance
- AWS::EC2::InternetGateway
- AWS::EC2::NetworkInterface
- AWS::IAM::AccessKey
- AWS::IAM::User
- OS::Ceilometer::Alarm
- OS::Cinder::Volume
- OS::Cinder::VolumeAttachment
- OS::Glance::Image
- OS::Heat::AccessPolicy
- OS::Heat::AutoScalingGroup
- OS::Heat::InstanceGroup
- OS::Heat::ScalingPolicy
- OS::Heat::Stack
- OS::Neutron::FloatingIP
- OS::Neutron::FloatingIPAssociation
- OS::Neutron::Net
- OS::Neutron::NetworkGateway
- OS::Neutron::Port
- OS::Neutron::Router
- OS::Neutron::RouterGateway
- OS::Neutron::RouterInterface
- OS::Neutron::SecurityGroup
- OS::Neutron::Subnet
- OS::Nova::Flavor
- OS::Nova::FloatingIP
- OS::Nova::FloatingIPAssociation
- OS::Nova::KeyPair
- OS::Nova::Server
- OS::Nova::ServerGroup
- OS::WR::ScalingPolicy

- WR::Neutron::PortForwarding
- WR::Neutron::ProviderNet
- WR::Neutron::ProviderNetRange
- WR::Neutron::QoSPolicy

Further Reading

Additional resources are available for understanding and working with OpenStack Heat Orchestration and templates.

The following online resources are suggested.

- *Create and manage stacks* (from the *OpenStack End User Guide*)
- *Heat* (from the OpenStack wiki)
- *Heat commands* (from the *OpenStack End User Guide*)
- *Heat Orchestration Template (HOT) Guide*
- *An Introduction to OpenStack Heat* (slide show from Mirantis)
- *AWS::CloudFromation::Init* (from Amazon's *AWS Documentation*; discusses metadata options for an EC2 instance)

Appendix A

Utilities for HP Helion OpenStack Carrier Grade

Topics:

HP Helion OpenStack Carrier Grade includes various utilities to help with system configuration and maintenance.

Utilities for vSwitch

You can query virtual switch (vSwitch) instances in HP Helion OpenStack Carrier Grade, and perform packet tracing on vSwitch interfaces.

In HP Helion OpenStack Carrier Grade, an accelerated virtual switch (AVS) runs on each compute host, providing Layer 2 switching for the virtual machines instantiated on the host. Up to 254 ports are supported for each host; this total includes physical, guest, and host ports.

You can use the `vshell` utility to query the status of each vSwitch and collect performance statistics. You can use the `vtrace` utility to perform packet trace operations.

Querying vSwitch

You can use the `vshell` command-line utility to query the internal state of the virtual switch on a compute host.

The `vshell` utility reports information for the vSwitch running on a compute host. This can be useful for debugging network-related issues.

You can run the utility from the active controller, or locally on the compute host. You must be logged in with root privileges.

To run `vshell`, use the following syntax:

```
$ sudo vshell [-H hostname] command
```

where

hostname

is the name of the host, if the utility is run from the controller.

command

is the vshell command to run.

The most common commands are listed below. For more information, use the following command:

```
$ sudo vshell help
```

Commonly used vshell commands

Note:

The sample outputs in this section have been truncated or otherwise modified to fit the available space.

engine-list

Lists details for the packet processing engine.

```
$ sudo vshell -H compute-1 engine-list
+---------+----+-------+-----------+-----------+-----------+
| uuid    | id | cpuid | socket-id | rxq-count | txq-count |
+---------+----+-------+-----------+-----------+-----------+
| dfcd... | 0  | 1     | 0         | 29        | 41        |
| ef55... | 1  | 2     | 0         | 28        | 41        |
+---------+----+-------+-----------+-----------+-----------+
```

port-list

Lists attributes for virtual and physical ports.

```
$ sudo vshell -H compute-1 port-list
+-------+----+-----------+-----------+-------------+-----------+------+
```

```
| uuid  | id | type      | socket-id | admin-state | link-state | mtu  |
+-------+----+-----------+-----------+-------------+------------+- ----+
| 21... | 0  | physical  | 0         | up          | up         | 1600 |
| d9... | 1  | physical  | 0         | up          | up         | 1600 |
| fd... | 2  | avp-guest | 0         | up          | up         | 1500 |
| da... | 3  | avp-guest | 0         | up          | up         | 1500 |
+-------+----+-----------+-----------+-------------+------------+------+

-------------+--------------+--------------+
 mac-address | network-uuid | network-name |
-------------+- ------------+--------------+
 90:e2:ba... |              |              |
 90:e2:ba... |              |              |
 fa:16:3e... | 4326a83a-... | net-4326a83a |
 fa:16:3e... | 4c02316e-... | net-4c02316e |
-------------+--------------+--------------+
```

interface-list

Lists attributes for logical interfaces.

```
$ sudo vshell -H compute-1 interface-list
+-------+----------+----------+-----------+-------------+------+
| uuid  | type     | class    | name      | mac-address | mtu  |
+-------+----------+----------+-----------+-------------+------+
| c6... | ae       | provider | ae0       | 90:e2:ba... | 1600 |
| 26... | ethernet | provider | eth0      | 90:e2:ba... | 1600 |
| d3... | vlan     | provider | ae0.632   | 90:e2:ba... | 1500 |
| 56... | vnic     | tenant   | vnic3     | fa:16:3e... | 1500 |
| a8... | vxlan    | provider | vxlan0    | 00:00:00... | 1500 |
+-------+----------+----------+-----------+-------------+------+

--------------+----------------+-------+
 network-uuid | network-name   | dvr   |
--------------+----------------+-------+
              |                | False |
              |                | False |
 8e0c9ba9-... | net-8e0c9ba9   | False |
 4326a83a-... | net-4326a83a   | False |
 582528d0-... | net-582528d0   | False |
--------------+----------------+-------+
```

lldp-neighbour-list

lists the learned Link Layer Discovery Protocol (LLDP) neighbors of physical ports.

```
$ sudo vshell -H compute-1 lldp-neighbour-list
+---------+-------------+----------------+-------------+--------+
| port-id | remote-port | remote-address | mac-address | rx-ttl |
+---------+-------------+----------------+-------------+--------+
| 0       | xe40        |                | 00:08:a2... | 120    |
| 1       | xe39        |                | 00:08:a2... | 120    |
+---------+-------------+----------------+-------------+--------+

-------------+--------------------+------------------+
 system-name | system-description | port-description |
-------------+--------------------+------------------+
 yow-cgcs... | yow-cgcs-quanta-1  | yow-cgcs-iron... |
 yow-cgcs... | yow-cgcs-quanta-1  | yow-cgcs-iron... |
-------------+--------------------+------------------+
```

network-table-list

Lists the learned and static MAC addresses of logical interfaces and network segments.

```
$ sudo vshell -H compute-1 network-table-list
+-------------+---------+----------------+---------------+--------------+
| mac-address | type    | interface-name | network-uuid  | network-name |
+------------ +---------+----------------+---------------+--------------+
| fa:16:3e... | dynamic | ae0.664        | 4326a83a-4... | net-4326a83a |
| fa:16:3e... | dynamic | ae0.632        | 8e0c9ba9-b... | net-8e0c9ba9 |
+-------------+---------+----------------+---------------+--------------+

-------------+-------------+
 created-age | timeout-age |
-------------+-------------+
 243552      | 2           |
 243656      | 5           |
-------------+-------------+
```

address-list

Lists IPv4 and IPv6 network addresses assigned to logical interfaces.

```
$ sudo vshell -H compute-1 address-list
+---------+----------------+----------------+----------------
+--------------------+
| address | interface-uuid | interface-name | valid-lifetime | preferred-
lifetime |
+---------+----------------+----------------+----------------
+--------------------+
| 192.... | c67943b7-d1... | ae0            | infinity       | infinity
        |
| fd00... | c67943b7-d1... | ae0            | infinity       | infinity
        |
| fe80... | 1e93938d-4b... | eth1           | infinity       | infinity
        |
| fe80... | 0198434f-f8... | ae0.680        | infinity       | infinity
        |
+---------+----------------+----------------+----------------
+--------------------+

-----------+----------+-------------+
 permanent | autoconf | floating-ip |
-----------+----------+-------------+
 True      | False    | False       |
 True      | False    | False       |
 False     | False    | False       |
 False     | False    | False       |
-----------+----------+-------------+
```

route-list

Lists the configured IPv4 and IPv6 network routes of virtual routers

```
$ sudo vshell -H compute-1 route-list

+--------+------------------+----------+----------------+----------------+
| router | prefix           | type     | interface-uuid | interface-name |
+--------+------------------+----------+----------------+----------------+
| 6b1... | 0.0.0.0/0        | external | c67943b7-d1... | ae0            |
| 6b1... | 192.168.60.64/27 | prefix   | c67943b7-d1... | ae0            |
| 6b1... | ::/0             | global   | c67943b7-d1... | ae0            |
```

```
| 6b1... | fd00:0:0:5::/64  | prefix   | c67943b7-d1... | ae0            |
+--------+------------------+----------+----------------+----------------+

---------------+--------+----------+-------------+-------------+
 gateway       | weight | lifetime | created-age | timeout-age |
---------------+--------+----------+-------------+-------------+
 192.168.60.65 | 1      | infinity | None        | None        |
 None          | 255    | infinity | None        | None        |
 fd00:0:0:5::1 | 1      | infinity | None        | None        |
 None          | 255    | infinity | None        | None        |
---------------+--------+----------+-------------+-------------+
```

ping

Initiate an ICMP echo request to a far-end IP address for connectivity testing

```
$ sudo vshell -H compute-1 ping ip-address source-interface
```

where

ip-address

is the far-end IP address

source-interface

is the UUID of the vSwitch interface to use

statistic-group-list

Lists available statistic groups. You can use these groups to enable or disable statistics collection for network-related activity. For performance reasons, the groups are disabled by default.

```
$ sudo vshell -H compute-1 statistic-group-list
+-----------+----------+
| name      | state    |
+-----------+----------+
| bridge    | disabled |
| dvr       | disabled |
| engine    | disabled |
| filter    | disabled |
| interface | disabled |
| ip        | disabled |
| layer2    | disabled |
| nat       | disabled |
| snat      | disabled |
| udp       | disabled |
| vxlan     | disabled |
+-----------+----------+
```

statistic-group-enable

Enables the statistic group specified by *groupname*. The following example enables the **engine** group:

```
$ sudo vshell -H compute-1 statistic-group-enable engine
```

Caution:

Real-time statistics collection can affect system performance. Always disable statistics collection when not in use.

statistic-group-disable

Disables the statistic group specified by *groupname*. The following example disables the **engine** group:

```
$ sudo vshell -H compute-1 statistic-group-disable engine
```

engine-stats-list

Displays packet processing and CPU utilization statistics for the packet processing engine. To collect fresh statistics, enable the **engine** statistic group.

```
$ sudo vshell -H compute-1 statistic-group-enable engine
$ sudo vshell -H compute-1 engine-stats-list
+-------+----+-------+------------+------------+-------------+
| uuid  | id | cpuid | rx-packets | tx-packets | tx-disabled |
+-------+----+-------+------------+------------+-------------+
| df... | 0  | 1     | 2536       | 7911       | 0           |
| ef... | 1  | 2     | 1446       | 5819       | 0           |
+-------+----+-------+------------+------------+-------------+

-------------+------------+------------+-------+
 tx-overflow | rx-discard | tx-discard | usage |
-------------+------------+------------+-------+
 0           | 0          | 0          | 0.00% |
 0           | 0          | 0          | 0.00% |
-------------+------------+------------+-------+
```

For best performance, disable the group when you are finished.

```
$ sudo vshell -H compute-1 statistic-group-disable engine
```

port-stat-list

Displays packet processing statistics by port. For this list, there is no corresponding statistic group. Collection is always enabled.

```
sudo vshell -H compute-1 port-stats-list
+-------+----+-----------+------------+------------+----------+----------+
| uuid  | id | type      | rx-packets | tx-packets | rx-bytes | tx-bytes |
+-------+----+-----------+------------+------------+----------+----------+
| 21... | 0  | physical  | 423017     | 154532     | 85008801 | 36865698 |
| fd... | 2  | avp-guest | 16444      | 101988     | 2147791  | 25666266 |
+-------+----+-----------+------------+------------+----------+----------+

-----------+-----------+-----------+
 tx-errors | rx-errors | rx-nombuf |
-----------+-----------+-----------+
 0         | 0         | 0         |
 0         | 0         | 0         |
-----------+-----------+-----------+
```

interface-stats-list

Displays packet processing statistics by interface. To collect fresh statistics, enable the **interface** statistic group.

```
$ sudo vshell -H compute-1 statistic-group-enable interface
$ sudo vshell -H compute-1 interface-stats-list
+-------+----------+-----------+------------+------------+----------
+----------+
| uuid  | type     | name      | rx-packets | tx-packets | rx-bytes | tx-
bytes |
```

```
+-------+----------+-----------+------------+------------+----------
+----------+
| c6... | ae       | ae0       | 0          | 0          | 0        | 0
     |
| 26... | ethernet | eth0      | 0          | 0          | 0        | 0
     |
| d3... | vlan     | ae0.632   | 0          | 0          | 0        | 0
     |
| 56... | vnic     | vnic3     | 0          | 0          | 0        | 0
     |
| a8... | vxlan    | vxlan0    | 0          | 0          | 0        | 0
     |
+-------+----------+-----------+------------+------------+----------
+----------+

-----------+-----------+-------------+-------------+-----------
+------------+
 tx-errors | rx-errors | tx-discards | rx-discards | rx-floods | rx-no-
vlan |
-----------+-----------+-------------+-------------+-----------
+------------+
 0         | 0         | 0           | 0           | 0         | 0
  |
 0         | 0         | 0           | 0           | 0         | 0
  |
 0         | 0         | 0           | 0           | 0         | 0
  |
 0         | 0         | 0           | 0           | 0         | 0
  |
 0         | 0         | 0           | 0           | 0         | 0
  |
-----------+-----------+-------------+-------------+-----------
+------------+
```

For best performance, disable the group when you are finished.

```
$ sudo vshell -H compute-1 statistic-group-disable interface
```

ip-stats-list

Displays packet processing statistics for the IP stack. To collect fresh statistics, enable the **ip** statistic group.

```
$ sudo vshell -H compute-1 statistic-group-enable ip
$ sudo vshell -H compute-1 ip-stats-list
+--------+-------+----------+--------------+--------------+
| family | cpuid | rx-total | rx-broadcast | rx-multicast |
+--------+-------+----------+--------------+--------------+
| ipv4   | 0     | 0        | 0            | 0            |
| ipv4   | 1     | 0        | 0            | 0            |
| ipv4   | 2     | 0        | 0            | 0            |
| ipv6   | 0     | 0        | 0            | 0            |
| ipv6   | 1     | 0        | 0            | 0            |
| ipv6   | 2     | 0        | 0            | 0            |
+--------+-------+----------+--------------+--------------+

----------+--------------+--------------+-----------+
 tx-total | tx-broadcast | tx-multicast | fwd-total |
----------+--------------+--------------+-----------+
 0        | 0            | 0            | 0         |
 0        | 0            | 0            | 0         |
 0        | 0            | 0            | 0         |
 0        | 0            | 0            | 0         |
 0        | 0            | 0            | 0         |
```

```
 0         | 0             | 0             | 0          |
-----------+---------------+---------------+------------+

----------------+----------------+--------------+---------------
+-----------------+
 rx-reassemble | tx-reassemble | tx-fragments | icmp-required | icmp-
throttled |
----------------+----------------+--------------+---------------
+-----------------+
 0              | 0              | 0            | 0             | 0
     |
 0              | 0              | 0            | 0             | 0
     |
 0              | 0              | 0            | 0             | 0
     |
 0              | 0              | 0            | 0             | 0
     |
 0              | 0              | 0            | 0             | 0
     |
 0              | 0              | 0            | 0             | 0
     |
----------------+----------------+--------------+---------------
+-----------------+
```

For best performance, disable the group when you are finished.

```
$ sudo vshell -H compute-1 statistic-group-disable ip
```

Performing Packet Tracing on vSwitch Interfaces

You can use the `vtrace` command to perform packet tracing on a vSwitch logical interface.

The **vtrace** command initiates a packet-trace capture on a logical interface. To use it, you must log on with root privileges to the host running the vSwitch instance.

Caution:

This command can cause significant packet loss on the trace port. Avoid using it on live systems with high rates of traffic.

When `vtrace` is used, all transmit and receive packets passing through the logical interface are mirrored to a dynamically created Linux host interface. The utility uses the vSwitch packet tracing API to enable tracing, and invokes **tcpdump** on the host interface. You can use any **tcpdump** option except -i to filter and analyze the captured data.

To initiate a packet trace, use the following command:

```
$ sudo vtrace interface-uuid tcpdump-options filter-expression
```

where

interface-uuid

is the UUID of the logical interface. You can obtain this using the `vshell interface-list` command.

tcpdump-options

are the options for the packet trace. For details, refer to the public documentation for **tcpdump**.

Note:

Do not use the -i option.

filter-expression

is an expression for filtering packet trace results. For details, refer to the public documentation for **tcpdump**.