Juniper Networks | Engineering Simplicity

Contrail Cloud v13 to v16 Upgrade Guide

Published
2022-01-14

Juniper Networks, Inc.
1133 Innovation Way
Sunnyvale, California 94089
USA
408-745-2000
www.juniper.net

Contrail Cloud v13 to v16 Upgrade Guide

The information in this document is current as of the date on the title page.

YEAR 2000 NOTICE

Juniper Networks hardware and software products are Year 2000 compliant. Junos OS has no known time-related limitations through the year 2038. However, the NTP application is known to have some difficulty in the year 2036.

END USER LICENSE AGREEMENT

The Juniper Networks product that is the subject of this technical documentation consists of (or is intended for use with) Juniper Networks software. Use of such software is subject to the terms and conditions of the End User License Agreement ("EULA") posted at https://support.juniper.net/support/eula/. By downloading, installing or using such software, you agree to the terms and conditions of that EULA.

Table of Contents

About This Guide | iv

1 Before You Begin

Prepare for the Upgrade | 2

- Upgrade Considerations | 2
- Request an Upgrade | 5
- During the Upgrade Process | 6

Understand the Upgrade Plan | 6

Contrail Cloud Upgrade Scripts | 10

2 Start the Upgrade

Update to the Latest Version of Contrail Cloud 13 | 15

Run the Pre-Upgrade Script | 15

Upgrade to Contrail Cloud v16.2 | 17

Migrate KVM to RHV | 20

3 Appendices

Configuration File Changes | 25

Sample Configuration Files v16.2 | 32

About This Guide

Use this guide to upgrade Contrail Cloud from latest version of Contrail Cloud 13 to Contrail Cloud Version 16.2. For a summary of upgrade enhancements, see: Contrail Cloud 16 Release Notes.

RELATED DOCUMENTATION

> https://www.juniper.net/documentation/en_US/contrail-cloud16/information-products/pathway-pages/contrail-cloud-deployment-guide.html

1

CHAPTER

Before You Begin

Prepare for the Upgrade | 2

Understand the Upgrade Plan | 6

Contrail Cloud Upgrade Scripts | 10

Prepare for the Upgrade

SUMMARY

This section describes important tasks to perform and upgrade considerations before upgrading to Contrail Cloud v16.2.

IN THIS SECTION

- Upgrade Considerations | 2
- Request an Upgrade | 5
- During the Upgrade Process | 6

Upgrade Considerations

IN THIS SECTION

- Check Your Cloud Health | 2
- Back up Your Undercloud and Overcloud | 2
- Pause and Shutdown Business Services | 3
- External NFS Storage | 3
- Undercloud Directory and Disk Space | 4

Check Your Cloud Health

Make sure that your cloud is functional, healthy, and that all services are active. Any problems in your cloud health can cause errors during the upgrade. To check your cloud health, see the Contrail Cloud Deployment Guide. Scroll to the "Verify Quorum and Node Health" section in the Node Reboot and Health Check appendix for instructions. See also, Validating Red Hat OpenStack Platform 13 before the upgrade.

Back up Your Undercloud and Overcloud

For back up instructions, see: Backup and Restore the Director Undercloud, and Backing up the Overcloud control plane services.

Pause and Shutdown Business Services

We recommend that you pause or shutdown external business services to ensure a smooth upgrade. Doing so helps prevent data loss or a workload error. The services are dependent on the specific business service or virtual machine (VM) that is running. It is important to know that there are no guarantees for the integrity of API calls during the upgrade. See the documentation for your specific service for instructions.

Do the following:

- Quiesce all external API requests, for example, Horizon.
- Perform a graceful shutdown on any vulnerable workloads.

External NFS Storage

You must provide an external NFS storage for the upgrade. Your NFS is used during the Contrail Cloud upgrade to back up the Contrail configuration database and the OpenStack controllers. External NFS storage is outside of the scope of Contrail Cloud and unique to your environment. Certain considerations need to be taken when preparing for an upgrade to Contrail Cloud 16.2. See below to understand how NFS storage functions with Contrail Cloud and the upgrade process.

- NFS storage must be provided to store backup information during the upgrade process.
- The Contrail configuration database and virtual machine images for the OpenStack overcloud controller role will be copied to the NFS mount.
- The NFS server must have sufficient space to hold the configuration database and the raw virtual machine image files. See the size calculation section below for more information.
- The NFS server must be reachable by the jump host and each control host, and the NFS server host name must be resolvable by their DNS. You configure DNS in the **config/site.yml** file under global['dns'] list.
- You must ensure that the NFS server authorizes export storage to the jump host and the control hosts.
    - Depending on the network topology for a particular cluster deployment, it can be common for the jump host to NAT traffic from the control hosts to destinations outside of the cluster. When using NAT, the NFS server sees traffic as originating from the jump host. You may not need explicit exports for the control hosts.
- Files on NFS will not be cleaned automatically after the upgrade. The files written to your NFS are kept for manual recovery purposes. You will need to remove these files manually after all steps have been completed.

Verify You Have NFS Connectivity

Verify that your jump host and control hosts can access your NFS share for the upgrade process.

Use the following CLI example on the jump host and each control host to verify you have a proper NFS path and permissions:

```
# Create the mount directory for the test
mkdir ~/nfs_test_mount
# Mount the share
sudo mount <ip/fqdn_of_nfs_server>:/<nfs_dir>  ~/nfs_test_mount
# Check if the share is properly mounted and permissions allow the host to create files
ls -al ~/nfs_test_mount
touch ~/nfs_test_mount/test-file
rm ~/nfs_test_mount/test-file
# Unmount and remove the mount directory
sudo umount ~/nfs_test_mount
rmdir ~/nfs_test_mount
```

NFS Storage Calculation

Make sure that you have enough external NFS storage before you upgrade. To determine how much storage that you need use the following calculation:

NFS size must be at least *X + (Y * N)* GB where:

X = The configuration database size.

Y = Is the configured image size from: `control_hosts['vm']['control']['disk']['vda']['size']` in the **config/site.yml** file.

N = The number of control hosts.

Undercloud Directory and Disk Space

During the upgrade process the undercloud virtual disk is adjusted to fit the requirements of RHEL 8.2. An automated backup of the VM is performed as part of this process. For this purpose, a dedicated directory with enough space is required on the jump host.

The default directory is "/home/{{ global['service_user']['name'] }}", but this can be changed in the "upgrade_plan" on page 6 by adding:

```
upgrade_plan:
  upgrade_backup_directory: <your_backup_directory>
```

Calculate the VM Disk Size

You need to adjust for the VM disk to meet the required minimal size. The size is a sum of fixed undercloud VM disk size and the current disk space consumed by the root partition on undercloud VM. You can access these values by using the following commands:

```
du -hs /var/lib/libvirt/images/undercloud.qcow2
```

and

```
ssh undercloud 'df -h /'
```

Request an Upgrade

Send an upgrade request to Juniper to have your key adjusted to Contrail Cloud 16. Juniper will adjust your unique key to now include two organizations - ContrailCloud/ContrailCloud16, or ContrailCloudNFR/ContrailCloud16NFR. Having both organizations registered with your unique key will allow the upgrade preparation software to be installed prior to the upgrade to Contrail Cloud 16.

Send an e-mail message to mailto:contrail_cloud_subscriptions@juniper.net and request a Contrail Cloud 16.2 upgrade. Provide the following information:

- Include your Contrail Cloud activation key in the email request. State that you are upgrading from v13 to v16.
- Specify the time and date for your upgrade.

  Juniper receives your request and activates your new upgrade key.

During the Upgrade Process

Do not make any configuration changes during the upgrade (such as enabling or disabling features, changing names, and so forth).

Do not attempt to change resources or cluster properties as part of the upgrade process. These activities should be handled as part of the deployment process, and is outside of the upgrade process. Making such changes during the upgrade process will likely lead to failures.

Understand the Upgrade Plan

IN THIS SECTION

- Configure Your Upgrade Plan | 7

This section provides information regarding the `upgrade_plan`, how it works, and how to configure the upgrade plan. A sample upgrade plan is provided to show a general example for how you might configure the `upgrade_plan`.

The upgrade plan controls the overcloud upgrade sequencing, and allows you to define node batches to upgrade when you run the **contrail-cloud-upgrade-overcloud-step2.sh script**. The `upgrade_plan` is defined in your **config/site.yml** configuration file.

The overcloud upgrade (**contrail-cloud-upgrade-overcloud-step2.sh**) will:

- Compare lock files to the user defined batch list to find the next user defined batch which has not already been completed.
- Upgrade all nodes per batch as defined in the upgrade plan nodes list, and is configured in the **config/site.yml** file.
- Upgrade packages and containers for each node defined in the current batch.
- Upgrade one batch per script run, as configured in the `upgrade_plan`.
- Automatically create a lock file when the batch has been processed. This is how the script knows to move on to the next batch.
- You rerun the script until there are no longer any batches left to upgrade.

Configure Your Upgrade Plan

The sample below shows the upgrade configuration options and is configured in the **config/site.yml** file. Use this upgrade configuration sample to determine accurate empty-space and indentations within the configuration hierarchy. Come back and reference this sample as needed to assist you through the upgrade process:

```
# Copyright 2021 Juniper Networks, Inc. All rights reserved.
# Licensed under the Juniper Networks Script Software License (the "License").
# You may not use this script file except in compliance with the License, which is located at
# http://www.juniper.net/support/legal/scriptlicense/
# Unless required by applicable law or otherwise agreed to in writing by the parties,
# software distributed under the License is distributed on an "AS IS" BASIS,
# WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
#
#
#

# The upgrade_plan provides the blueprint for working through the deployment upgrade.
# The plan is defined as a series of "batches".  Each batch is processed by the
# contrail-cloud-upgrade-overcloud-step2.sh, one batch for each run.  Once a batch is
# successfully completed a lock file is created to indicate that state.  The next run
# of the contrail-cloud-upgrade-overcloud-step2.sh script will find the next unfinished
# batch to process.
upgrade_plan:
  # batches is an array of steps necessary to complete the full upgrade.
  # Each batch defines:
  #   name: a unique batch name (used for lock file naming)
  #   upgrade_type: either "sequence" (one node at a time) or
  #                         "parallel" (all nodes together)
  #   nodes_list: a list of nodes to apply the upgrade to.  The special
  #               keyword "ControlPlane" represents the Openstack and Contrail
  #               control plane roles.  For each of these roles, the upgrade
  #               will be performed on a single instance at a time sequentially
  #               until all roles instances are upgraded.
  batches:
  # The first batch should always be the control plane.
  - name: controlplane
    upgrade_type: sequence
    nodes_list:
    - ControlPlane
```

```
  # This batch is used to upgrade all the DPDK and kernel computes in rack0 in parallel.
  - name: computenodes_rack0
    upgrade_type: parallel
    nodes_list:
    - overcloudx51-compdpdk0hw2-0
    - overcloudx51-compkernel0hw0-0
    - overcloudx51-compkernel0hw1-0
  # This batch is used to upgrade all the DPDK and kernel computes in rack1 in parallel.
  - name: computenodes_rack1
    upgrade_type: parallel
    nodes_list:
    - overcloudx51-compdpdk1hw3-0
    - overcloudx51-compkernel1hw0-0
    - overcloudx51-compkernel1hw1-0
  # This batch is used to upgrade all the SRIOV computes in sequence (just an example).
  - name: sriovnodes
    upgrade_type: sequence
    nodes_list:
    - overcloudx51-compsriov0hw4-0
    - overcloudx51-compsriov1hw5-0
  # Batch to upgrade ceph storage nodes.  To ensure the integrity of the ceph cluster,
  # each role instance will be upgraded sequentially.
  - name: cephnodes
    upgrade_type: sequence
    nodes_list:
    - overcloudx51-cephstorage0hw6-0
    - overcloudx51-cephstorage0hw6-1
    - overcloudx51-cephstorage1hw7-0
  # Path to a storage share that will be mounted with NFS and used for backup.
  # This NFS share will be mounted by the jumphost and control hosts during the upgrade.
  nfs_path: "nfs.my-domain.com:/nfs"
```

The procedure below allows you to target specific nodes during the upgrade. This approach allows for control and predictability of the upgrade. This method is desirable so you can target specific resources to be upgraded as workloads are migrated.

To complete a targeted upgrade, edit the sample plan to match your deployment for each targeted batch. Later in the procedure you will run the `contrail-cloud-upgrade-overcloud-step2.sh` script for each batch defined within the upgrade plan. Add the `upgrade_plan:` configuration to your **config/site.yml** configuration file.

Name Your Unique Batch

The upgrade will be performed in the batch order you configure in your `upgrade_plan`. The unique batch name is used to identify each unique batch, and for lock file naming.

When the `contrail-cloud-upgrade-overcloud-step2.sh` script is run, the script will identify the first unique named batch which has not already been executed and upgraded. A lockfile is created after each successful batch upgrade to identify it as being completed. To start, you might configure it to look like this:

```
upgrade_plan:

  batches:
  # unique batch name
  - name: controlplane
```

Define the Upgrade Type for the Named Node Role

The upgrade type determines the order in which the nodes are upgraded in the named batch run. You have the option of running the upgrade in sequence and will upgrade one node at a time, per batch run. You also have the option to run the upgrade in parallel and will upgrade all nodes together, per batch run.

The sample below is configured to upgrade the nodes in the named batch in sequence:

```
upgrade_plan:

  batches:
    #   upgrade_type: either "sequence" (one node at a time) or
    #                         "parallel" (all nodes together)
    upgrade_type: sequence
```

Define the nodes in the Named Batch

Define the nodes in the `nodes_list` that you would like to upgrade in the named batch run. These nodes are the specific node names from your environment.

ControlPlane is a special keyword understood by the upgrade plan. The ControlPlane represents the OpenStack control plane role (Controller), and the Contrail control plane roles (ContrailController, ContrailAnalytics, and ContrailAnalyticsDatabase). The ControlPlane will upgrade as one batch as part of the upgrade plan. One instance of each role will be upgraded in parallel, and then the next instances will be run in parallel, until all roles are upgraded. Within each role, only one instance of the role will be upgraded at a time.

The first batch in your `upgrade_plan` should always be the ControlPlane role:

```
upgrade_plan:

  batches:
    #   nodes_list: a list of nodes to apply the upgrade to.  The special
    #               keyword "ControlPlane" represents the Openstack and Contrail
    #               control plane roles.  For each of these roles, the upgrade
    #               will be performed on a single instance at a time sequentially
    #               until all roles instances are upgraded.
    nodes_list:
    - ControlPlane
```

Finish Defining Your Remaining Batches

Define the rest of your batches and ensure all nodes have been targeted for upgrade. The ControlPlane (as a set of control plane nodes) should always be the first batch to upgrade. The next batches should include the compute nodes. The last batches should consist of the storage nodes.

Contrail Cloud Upgrade Scripts

SUMMARY

The table below describes the scripts you use in the upgrade procedure. The script files all reside in the **/var/lib/contrail_cloud/scripts** directory. You should include " `-d`" after every script invocation . This will capture additional debug information during the script run, and is helpful when troubleshooting.

Table 1: Contrail Cloud Upgrade Scripts

Script Name	Description
pre-contrail-cloud-upgrade.sh -d	• Prepares all nodes in the environment for upgrade: the jump host, undercloud, and overcloud nodes. • Installs the Leapp package used to upgrade RHEL7 to RHEL8. • Organizes your SSL certificates into a new directory structure
contrail-cloud-upgrade-jumphost.sh -d	• Upgrades the jump host to use RHEL 8.2.
contrail-cloud-upgrade-undercloud.sh -d	• Upgrades the undercloud to use RHEL 8.2. • Upgrades the packages and containers on the undercloud VM. • Publishes new containers to the registry on the jump host. • Upgrades Red Hat OpenStack Platform Director on the undercloud VM. • Rebuilds the overloud-full image used to provision all new overcloud role instances.
pre-contrail-cloud-upgrade-overcloud.sh -d	• Sets the clusters into maintenance mode, prepares new scripts for reregistration, and upgrades the network configuration.
contrail-cloud-upgrade-overcloud-step1.sh -d	• Upgrades the overcloud plan on the undercloud VM. • Prepares the overcloud heat stack for the upgrade: openstack overcloud upgrade prepare

Table 1: Contrail Cloud Upgrade Scripts *(Continued)*

Script Name	Description
contrail-cloud-upgrade-overcloud-step2.sh -d	• Upgrades all nodes in the current batch as defined in the site.yml configuration file under the upgrade_plan. • Upgrades packages and containers for each node batch, per batch run. • Upgrades one batch per script run. • Processes the batch file and automatically creates a lockfile. • Every role will be processed in order per the defined batch run.
contrail-cloud-upgrade-overcloud-step3.sh -d	• Runs the upgrade converge: openstack overcloud upgrade converge. • When Ceph is enabled – upgrades and runs the Ceph cluster configuration: openstack overcloud external-upgrade run -y --stack overcloud --tags ceph. • Finalizes the overcloud upgrade.
install_rhvm.sh -d	• Deploys RHVM on the jump host. • Configures RHVM.
pre-contrail-cloud-upgrade-kvm2rhv.sh -d	• Performs maintenance of the VMs, and sets up NFS.

Table 1: Contrail Cloud Upgrade Scripts *(Continued)*

Script Name	Description

contrail-cloud-upgrade-kvm2rhv.sh -d	• Performs backup of the control host VMs.
	• Redeploys the control hosts and is run once for every control host instance.
	Each control host is completely torn down and rebuilt with new VMs and new VM storage.
	• Deploys from the backup:
	  • The control VM which hosts the control overcloud role.
	  • Creates the appformix-controller VM and adds it to Ironic.

post-contrail-cloud-upgrade-kvm2rhv.sh -d	• Creates the contrail-k8s VM and adds it to Ironic. • Performs post-migration cleanup, turns maintenance off, and refreshes facts and Ironic configuration.
k8s-cluster-deploy.sh -d	Deploys the Kubernetes cluster onto the contrail-k8s VMs on each control host. This will host the Contrail control plane
k8s-tf-operator-deploy.sh -d	• Deploys the Contrail control plane. • Restores the configuration database from backup in NFS storage
contrail-insights-deploy.sh -d	Installs Contrail Insights (formerly AppFormix) onto the appformix-controller VMs on each control host.

2
CHAPTER

Start the Upgrade

Update to the Latest Version of Contrail Cloud 13 | 15

Run the Pre-Upgrade Script | 15

Upgrade to Contrail Cloud v16.2 | 17

Migrate KVM to RHV | 20

Update to the Latest Version of Contrail Cloud 13

The upgrade requires that you have the latest version of Contrail Cloud 13. If you are not running the latest Contrail Cloud version, follow the upgrade instructions in the Contrail Cloud 13 Release Notes.

Verify that your configuration files in the **/var/lib/contrail_cloud/config directory** are accurate for your Contrail Cloud deployment.

1. Specify the activation key by setting the environment variables.
   For example:

```
SATELLITE="contrail-cloud-satellite.juniper.net"
SATELLITE_KEY="ak-my-account-key"
SATELLITE_ORG="ContrailCloud"
```

2. Run the installer script (`contrail_cloud_installer.sh`) from the jump host CLI with your environment variables in the example below. The script will install the latest Contrail Cloud packages for your Contrail Cloud 13 release. The packages are installed in the **/var/lib/contrail_cloud** directory.

   NOTE: This is the installer that came with Contrail Cloud 13. This is not the installer that comes with your Contrail Cloud 16 upgrade.

```
./contrail_cloud_installer.sh\
--satellite_host ${SATELLITE} \
--satellite_key ${SATELLITE_KEY} \
--satellite_org ${SATELLITE_ORG}
```

Run the Pre-Upgrade Script

The pre-upgrade script prepares the jump host, undercloud, and overcloud nodes for the upgrade. The script installs the Leapp package which is used to upgrade RHEL7 to RHEL8, and organizes your SSL certificates into a new directory structure.

1. As the contrail user, run the following script from the jump host:

```
/var/lib/contrail_cloud/scripts/pre-contrail-cloud-upgrade.sh -d
```

Do not make any configuration changes during the upgrade (such as enabling or disabling features, changing names, and so forth). Do not attempt to change resources or cluster properties as part of the upgrade process.

> **NOTE**: Do not make any configuration changes to your **overcloud-nics.yml** file. Configuration changes to your **overcloud-nics.yml** file should be applied before the "Migrate KVM to RHV" on page 20 section.
>
> The VM interface names in the **overcloud-nics.yml** file must use the emX interface naming.

2. Update your configuration files for v16.2 as specified in the "Configuration File Changes" on page 25 appendix.

   Additional NIC changes will need to be made, and focus in the areas below:

   a. Your VM interface naming in the **overcloud-nics.yml** file must not use ethX naming. This must be changed before starting the upgrade to Contrail Cloud 16.2.

   b. You must ensure that the VM interface naming in the **overcloud-nics.yml** file use emX naming structure. These interfaces include:

      - `Controller_network_config`
      - `Contrail_network_config`
      - `AppformixController_network_config`

      When defining interfaces:

      - Strictly define all interfaces, even interfaces you do not use.
      - Do not use DHCP, as shown in the example above.
      - You will update the interface naming in two steps: prior to the upgrade (now), and prior to "Migrate KVM to RHV" on page 20.

      Your change will look similar to the example below:

```
- type: interface
   name: em2
   use_dhcp: false
```

c. If your environment contains sriov computes, verify if **/etc/sysconfig/network-scripts/ifcfg-ensXnpY** exists on these systems with sriov. If the file exists, this information should also be defined in the **overcloud_nics.yml** configuration file in the same way other unused interfaces are strictly defined. The example below defines an sriov interface:

```
- type: interface
  name: ens1np0
  use_dhcp: false
  onboot: false
```

Ensure that all EC2Metadata routes are removed. See the example below for the section of configuration to be removed in the **overcloud_nics.yml** file:

```
    ip_netmask: /32
    next_hop:
      get_param: EC2MetadataIp
```

Upgrade to Contrail Cloud v16.2

The procedure below guides you through the upgrade to Contrail Cloud v16.2. This delivers upgraded Contrail containers, Red Hat RHEL/RHOSP/storage content, and kernel version.

NOTE: There will be a major disruption in service during the upgrade. Whole services will be taken down during the upgrade process. Plan for a long period of disruption for the upgrade process to complete. The upgrade preserves existing overcloud configurations. For example: images, projects, networks, volumes, virtual machines, and so on.

As the contrail user, (**su - contrail** from root), run the following scripts in the CLI from the jump host:

1. Upgrade the Jump host
   Run the `contrail-cloud-upgrade-jumphost.sh` script to upgrade the jump host from RHEL 7.9 to 8.2.

The `contrail-cloud-upgrade-jumphost.sh` script run is interrupted during the leapp upgrade. The jump host reboots as part of the leapp upgrade process, and you must rerun the script after each subsequent reboot:

```
/var/lib/contrail_cloud/scripts/contrail-cloud-upgrade-jumphost.sh -d
```

Perform the following steps after the reboot from the previous script run:

a. Enter the `uname -a` command to check that the kernel information is correct. From the jump host CLI:

```
uname -a
```

b. Enter the `cat /etc/redhat-release` command to verify that your RHEL version is now RHEL v8.2. From the jump host CLI:

```
cat /etc/redhat-release
```

c. Rerun the script for a second time after the jump host reboot and version verification:

```
/var/lib/contrail_cloud/scripts/contrail-cloud-upgrade-jumphost.sh -d
```

This will:

- Update the packages and containers on the jump host.

d. Rerun the script for a third and final time after the jump host reboots:

```
/var/lib/contrail_cloud/scripts/contrail-cloud-upgrade-jumphost.sh -d
```

This will:

- perform a yum cleanup on the jump host.

2. Upgrade the Undercloud
Run this script to upgrade the undercloud virtual machine (VM) on the jump host. This process also prepares the undercloud for the overcloud upgrade, pulls new images, and redeploys the undercloud services into containers.

```
/var/lib/contrail_cloud/scripts/contrail-cloud-upgrade-undercloud.sh -d
```

This will:

- Update the packages and containers on the undercloud VM.
- Update Red Hat OpenStack Platform Director on the undercloud VM, and organize registry with overcloud containers.
- Update image on the undercloud VM used to provision all new overcloud role instances.
  - overcloud-image-full is upgraded and used to provision any new overcloud role instance.

3. Prepare Contrail Cloud for the upgrade.
   Run this script to upgrade your network configurations, prepare new scripts for reregistration, and prepare the RHOSP control plane for upgrade.

```
/var/lib/contrail_cloud/scripts/pre-contrail-cloud-upgrade-overcloud.sh -d
```

   This will:

   - Disable STONITH.
   - Resync Leapp files.
   - Applies fixes for the overcloud which align node configuration for upgrade.

4. Prepare the Overcloud and RHOSP control plane for the upgrade.
   Run the script to prepare the RHOSP control plane and OpenStack overcloud for upgrade.

```
/var/lib/contrail_cloud/scripts/contrail-cloud-upgrade-overcloud-step1.sh -d
```

   This will:

   - Runs `openstack overcloud upgrade prepare` to generate the necessary Ansible configurations for overcloud upgrade.

5. Run the overcloud batch upgrades.
   Run the upgrade script for each named batch until all batches have successfully upgraded and there are no longer any named batches left to process. The script will not automatically go through all named batches in one run. You must manually run the upgrade script for each unique named batch.

```
/var/lib/contrail_cloud/scripts/contrail-cloud-upgrade-overcloud-step2.sh -d
```

   The overcloud upgrade (**contrail-cloud-upgrade-overcloud-step2.sh**) will:

   - Upgrade all nodes as defined in the `upgrade_plan['batches'][<batch>]['nodes_list']` and is configured in the **config/site.yml** file.

- Upgrade node packages and containers according to the batch `upgrade_type`.
  - Parallel = all at the same time.
  - Sequence = one at a time
- Upgrade one batch per script run, as configured in the `upgrade_plan`.
- Automatically create a lockfile when the batch has been processed. This is how the script knows to move on to the next batch the next time the script is run.
- You rerun the script until there are no longer any batches left to upgrade.

6. Converge the Overcloud Upgrade
   Run this script to converge the overcloud heat stack, and then upgrade Ceph from Version 3 to Version 4 (if Ceph is enabled).

```
/var/lib/contrail_cloud/scripts/contrail-cloud-upgrade-overcloud-step3.sh -d
```

   This will:

   - Ensure that the stack resource structure aligns with the new packages and configurations.
   - Converge the overcloud and runs `openstack overcloud upgrade converge` to complete the task.
   - Upgrade the Ceph cluster configuration and runs `openstack overcloud external-upgrade run -tags ceph` to complete the task.
   - Finalize the overcloud update.

RELATED DOCUMENTATION

Contrail Cloud Upgrade Scripts | 10

Migrate KVM to RHV

Starting in v16.2, Contrail Cloud now uses Red Hat Enterprise Virtualization (RHEV, or RHV) for the control host hypervisors. The jump host and overcloud computes will continue to use Kernel-based Virtual Machine (KVM).

Follow these steps to migrate the control host KVM to RHV:

NOTE: Configuration changes to your **overcloud-nics.yml** file must be made at this point.

You must change the VM interface naming in your **overcloud-nics.yml** file before starting the steps below. The VM interface naming must be changed from `emX` to the `enp(x+1)s0` naming structure. These interfaces include:

- Controller_network_config
- AppformixController_network_config

Your change will look similar to the example below:

```
- type: interface
  name: enp1s0
  use_dhcp: false
```

You can reference the samples provided in "Configuration File Changes" on page 25 for more details.

1. Deploy Red Hat Virtualization Manager (RHVM) from the jump host.

   ```
   /var/lib/contrail_cloud/scripts/install_rhvm.sh -d
   ```

2. Log in to the RHV Manager Web UI to verify that RHVM deployed successfully. Your ability to access the web UI is verification of a successful RHV Manager deployment. To access the GUI, enter **https://<jumphost fqdn>/ovirt-engine/** in your browser. The default user is `admin`, and the default password is `c0ntrail123`.
3. Test NFS and back up the contrail configuration database onto the NFS storage.

   ```
   /var/lib/contrail_cloud/scripts/pre-contrail-cloud-upgrade-kvm2rhv.sh -d
   ```

4. Migrate your control host. This will migrate a single node per script run.

   ```
   /var/lib/contrail_cloud/scripts/contrail-cloud-upgrade-kvm2rhv.sh -d
   ```

   This script will:

   - Perform the maintenance and backup of the openstack controller VM.
   - Re-image and reconfigure the control host.

- Connects the VM to the RHVM and sets up VM storage.
- Creates the OpenStack VM, k8s_host VM, and Contrail Insights VM and adds them to Ironic as resources.
- Deploys the OpenStack VM from the backup.

5. Run the upgrade script for each remaining control host, until all control hosts have been converted.

```
/var/lib/contrail_cloud/scripts/contrail-cloud-upgrade-kvm2rhv.sh -d
```

6. Run the post-migration script to perform cleanup, refresh Ironic configuration and facts, and disable maintenance mode.

```
/var/lib/contrail_cloud/scripts/post-contrail-cloud-upgrade-kvm2rhv.sh -d
```

7. Deploy the Kubernetes (k8s) clusters.

```
/var/lib/contrail_cloud/scripts/k8s-cluster-deploy.sh -d
```

This will:

- Deploy RHEL8 onto the k8s_host VMs in each control host.
- Deploy a Kubernetes cluster onto the set of k8s_host VMs.

8. Find your overcloud stack ID.
You will need your overcloud stack ID for setting up your NFS backup directory in the steps below. Run the following from the jump host:

```
ssh undercloud "source stackrc; openstack stack show overcloud -f value -c id"
```

9. Mount the NFS backup directory on jump host.

```
mkdir -p /var/lib/contrail_cloud/nfs_mount
sudo mount <ip/fqdn_of_nfs_server>:/<nfs_dir> /var/lib/contrail_cloud/nfs_mount
ls -al /var/lib/contrail_cloud/nfs_mount/<your_overcloud_stack_id>/db-dump.json
```

10. Deploy the Contrail cluster.

```
/var/lib/contrail_cloud/scripts/k8s-tf-operator-deploy.sh -d -i /var/lib/contrail_cloud/
nfs_mount/<your_overcloud_stack_id>/db-dump.json
```

11. Unmount and remove your NFS directory.

```
sudo umount /var/lib/contrail_cloud/nfs_mount && rmdir /var/lib/contrail_cloud/nfs_mount
```

12. Run the overcloud deployment to update OpenStack with the new Contrail virtual IP address values.

```
/var/lib/contrail_cloud/scripts/openstack-deploy.sh -d
```

13. Run Contrail Insight deploy (if required for your environment).

```
/var/lib/contrail_cloud/scripts/contrail-insights-deploy.sh -d
```

14. It is possible that the upgrade could interrupt some of the PCS (Pacemaker) resources. From one of the OpenStack controllers, run the following command to verify all PCS resources were properly started:

```
sudo pcs status
```

    a. Make sure any failed resources are started and stable. Cleanup any failed actions by running the following command:

    ```
    sudo pcs resource cleanup
    ```

15. Verify all Contrail networks, ports, policies, and configurations are correct. You can now manually remove the files from the NFS directory after you have verified a successful upgrade.

```
<ip/fqdn_of_nfs_server>:/<nfs_dir>/<your_overcloud_stack_id>
```

3

CHAPTER

Appendices

Configuration File Changes | 25

Sample Configuration Files v16.2 | 32

Configuration File Changes

IN THIS SECTION

- v13 to v16 Configuration File Changes | 25

This appendix describes the changes you need to make to your configuration files for the upgrade from Contrail Cloud v13.6 to Contrail Cloud v16.2.

NOTE: The configuration files shown here only contain snippets of what's changed fromv13.6 to v16.2. For the complete configuration, see: "Sample Configuration Files v16.2" on page 32.

v13 to v16 Configuration File Changes

Make the following changes to your YAML configuration files during the "Prepare for the Upgrade" on page 2 section of this guide.

1. In your **site.yml** configuration file, copy and paste the `upgrade_plan` section. Change the configuration variables to match your specific deployment environment:

```
# Copyright 2021 Juniper Networks, Inc. All rights reserved.
# Licensed under the Juniper Networks Script Software License (the "License").
# You may not use this script file except in compliance with the License, which is located at
# http://www.juniper.net/support/legal/scriptlicense/
# Unless required by applicable law or otherwise agreed to in writing by the parties,
# software distributed under the License is distributed on an "AS IS" BASIS,
# WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
#
#
#

# The upgrade_plan provides the blueprint for working through the deployment upgrade.
# The plan is defined as a series of "batches".  Each batch is processed by the
# contrail-cloud-upgrade-overcloud-step2.sh, one batch for each run.  Once a batch is
```

```
# successfully completed a lock file is created to indicate that state.  The next run
# of the contrail-cloud-upgrade-overcloud-step2.sh script will find the next unfinished
# batch to process.
upgrade_plan:
  # batches is an array of steps necessary to complete the full upgrade.
  # Each batch defines:
  #   name: a unique batch name (used for lock file naming)
  #   upgrade_type: either "sequence" (one node at a time) or
  #                         "parallel" (all nodes together)
  #   nodes_list: a list of nodes to apply the upgrade to.  The special
  #               keyword "ControlPlane" represents the Openstack and Contrail
  #               control plane roles.  For each of these roles, the upgrade
  #               will be performed on a single instance at a time sequentially
  #               until all roles instances are upgraded.
  batches:
  # The first batch should always be the control plane.
  - name: controlplane
    upgrade_type: sequence
    nodes_list:
    - ControlPlane
  # This batch is used to upgrade all the DPDK and kernel computes in rack0 in parallel.
  - name: computenodes_rack0
    upgrade_type: parallel
    nodes_list:
    - overcloudx51-compdpdk0hw2-0
    - overcloudx51-compkernel0hw0-0
    - overcloudx51-compkernel0hw1-0
  # This batch is used to upgrade all the DPDK and kernel computes in rack1 in parallel.
  - name: computenodes_rack1
    upgrade_type: parallel
    nodes_list:
    - overcloudx51-compdpdk1hw3-0
    - overcloudx51-compkernel1hw0-0
    - overcloudx51-compkernel1hw1-0
  # This batch is used to upgrade all the SRIOV computes in sequence (just an example).
  - name: sriovnodes
    upgrade_type: sequence
    nodes_list:
    - overcloudx51-compsriov0hw4-0
    - overcloudx51-compsriov1hw5-0
  # Batch to upgrade ceph storage nodes.  To ensure the integrity of the ceph cluster,
  # each role instance will be upgraded sequentially.
  - name: cephnodes
```

```
    upgrade_type: sequence
    nodes_list:
    - overcloudx51-cephstorage0hw6-0
    - overcloudx51-cephstorage0hw6-1
    - overcloudx51-cephstorage1hw7-0
  # Path to a storage share that will be mounted with NFS and used for backup.
  # This NFS share will be mounted by the jumphost and control hosts during the upgrade.
  nfs_path: "nfs.my-domain.com:/nfs"
```

2. In your **control-host.yml** file, configure the interfaces section with the following snippet. The configurations include updates to the interface type (ovs-bridges to standard interfaces) and interface network names (emX to enp(x+1)s0).

NOTE: Do not change the order of the interfaces.

```
  - type: interface
    name: eno1
    nm_controlled: true
    use_dhcp: true
  - type: linux_bond
    name: bond0
    nm_controlled: true
    use_dhcp: false
    bonding_options: "mode=802.3ad xmit_hash_policy=layer3+4 lacp_rate=fast updelay=1000
miimon=100"
    members:
      -
        name: ens7f0
        type: interface
        use_dhcp: false
        nm_controlled: true
        primary: true
      -
        name: ens7f1
        type: interface
        use_dhcp: false
        nm_controlled: true

control_hosts:
  vm_interfaces:
```

```
    - interface: enp1s0
      physical_interface: eno1
    - interface: enp2s0
      physical_interface: eno2
    - interface: enp3s0
      physical_interface: bond0
```

3. Strictly define all interfaces, even interfaces that you don't use, to not use DHCP as shown in this example. The **config/overcloud-nics.yml** interfaces must match the interface names provided in the **config/control-hosts.yml**`vm_interfaces` configuration.

```
- type: interface
    name: nic2
    use_dhcp: false
```

4. In your **vault-data.yml** file (using ansible-vault-edit). Add configuration changes which include the new RHVM configuration.

```
vault:

  rhvm:
    vm:
      # rhvm user name
      user: "contrail"
      # password for the rhvm vm user
      password: "c0ntrail123"
      # root password for the rhvm VM
      root_password: "c0ntrail123"
      # keystone admin password
      admin_password: "c0ntrail123"
      # Passphrase used to encrypt ssh key of rhvm user.
      # If not defined ssh private key will not be encrypted.
      # ssh_key_passphrase: "c0ntrail123"
      vnc:
        # VNC console password for the rhvm VM
        password: "contrail123"
```

5. In your **site.yml** configuration file, remove any unused VM definitions as shown in the example below.

```
control_hosts:
  vm:
    contrail-analytics:
    ...
    contrail_controller:
    ...
    contrail_anaylitcs_database
    ...
```

6. In your **site.yml** configuration file, add your new organization for Contrail Cloud 16:

```
global:

  rhel:
    # Contrail Cloud Activation Key
    # These details are provided when you request an activation key from
    # contrail cloud subscriptions <contrail_cloud_subscriptions@juniper.net>
    #
    satellite:
      #SATELLITE_KEY should be defined in vault-data.yml file
      #SATELLITE_ORG
      organization: "ContrailCloud16"
      #SATELLITE_FQDN
      fqdn: contrail-cloud-satellite.juniper.net
```

7. In your **site.yml** configuration file, Update your storage configuration to also include mount point for control-hosts storage volumes in Contrail Cloud 16:

```
control_hosts:
  storage:

ssd_storage:

      mountpoint: "/srv/
ssd"

      type:
```

```
logical


disk:

        - "/dev/sdb"
```

8. In your **site.yml** configuration file, apply all needed `extra_configs` for computes as per **samples/features/extra-config/site.yml**:

```
# Extra config allows to provide any Heat variable (including those not used by ContrailCloud)

overcloud:
  extra_config:
    # Configure NetworkDeploymentActions to always re-run os-net-config
    # NetworkDeploymentActions: ['CREATE','UPDATE']
    ContrailDpdkOptions: "--vr_mempool_sz 65536 --dpdk_txd_sz 2048 --dpdk_rxd_sz 2048"
    ComputeDpdkParameters:
      # ComputeDpdk specific parameters which will override the defaults
      # ComputeDpdk - use amd_iommu if using AMD CPU
      KernelArgs: "amd_iommu=on iommu=pt default_hugepagesz=1GB hugepagesz=1G hugepages=64
hugepagesz=2M hugepages=2048 isolcpus=2-9,22-29"
      ContrailDpdkHugepages1GB: 64
      ContrailDpdkHugepages2MB: 2048
      IsolCpusList: "2-9,22-29"
      NovaVcpuPinSet: ['4-9','24-29']
      ExtraSysctlSettings:
        vm.nr_hugepages:
          value: 64
        vm.max_map_count:
          value: 128960
    ComputeKernelParameters:
      # ComputeKernel specific parameters which will override the defaults
      # ComputeKernel - use intel_iommu if using Intel CPU
      KernelArgs: "intel_iommu=on iommu=pt default_hugepagesz=1GB hugepagesz=1G hugepages=64
hugepagesz=2M hugepages=2048"
      ContrailVrouterHugepages1GB: 64
      ContrailVrouterHugepages2MB: 2048
      ExtraSysctlSettings:
        vm.nr_hugepages:
          value: 64
```

```
        vm.max_map_count:
          value: 128960
    NovaSchedulerDefaultFilters:
      - RetryFilter
      - AvailabilityZoneFilter
      - RamFilter
      - DiskFilter
      - ComputeFilter
      - ComputeCapabilitiesFilter
      - ImagePropertiesFilter
      - ServerGroupAntiAffinityFilter
      - ServerGroupAffinityFilter
      - AggregateInstanceExtraSpecsFilter
      - NUMATopologyFilter
    # Note: applied to compute and dpdkcompute
    NovaVcpuPinSet: ['4-11','28-35','13-23','37-47']
    NovaComputeExtraConfig:
      nova::cpu_allocation_ratio: 1.0
      nova::ram_allocation_ratio: 1.0
      nova::disk_allocation_ratio: 1.0
    ContrailDpdkExtraConfig:
      nova::cpu_allocation_ratio: 1.0
      nova::ram_allocation_ratio: 1.0
      nova::disk_allocation_ratio: 1.0
    ControllerExtraConfig:
      nova::config::nova_config:
        filter_scheduler/build_failure_weight_multiplier:
          value: 10000.0
```

```
overcloud:
  extra_config:
    ComputeDpdk2Hw2Parameters:
      KernelArgs: "intel_iommu=on iommu=pt default_hugepagesz=1GB hugepagesz=1G hugepages=64
hugepagesz=2M hugepages=2048"
      ContrailDpdkOptions: "--vr_mempool_sz 131072 --dpdk_txd_sz 2048 --dpdk_rxd_sz 2048 --
vr_flow_entries=4000000"
    ComputeKernel0Hw0Parameters:
      KernelArgs: "intel_iommu=on iommu=pt default_hugepagesz=1GB hugepagesz=1G hugepages=64
hugepagesz=2M hugepages=2048"
      ContrailVrouterHugepages1GB: 64
      ContrailVrouterHugepages2MB: 2048
```

```
    ComputeSriov1Hw3Parameters:
      KernelArgs: "intel_iommu=on iommu=pt default_hugepagesz=1GB hugepagesz=1G hugepages=64
hugepagesz=2M hugepages=2048"
      ContrailSriovHugepages1GB: 64
      ContrailSriovHugepages2MB: 2048
```

9. Add the K8s configuration in your **site.yml** configuration file. For more information about configuring K8s, see the Contrail Cloud Deployment Guide for v16.

```
control_hosts:

  vm:
    # VMs for ContrailController role
    contrail-k8s:
      disk:
        # Root disk
        vda:
          # Virsh storage pool (see storage above)
          pool: hdd_storage


k8s:
  external_vip_ip:
  internal_api_vip_ip:
```

RELATED DOCUMENTATION

Upgrade to Contrail Cloud v16.2 | 17

Sample Configuration Files v16.2

IN THIS SECTION

- Sample site.yml Configuration File | 33
- Sample inventory.yml Configuration File | 57

- Sample control-host-nodes.yml Configuration File | 59
- Sample overcloud-nics.yml Configuration File | 61
- Sample compute-nodes.yml Configuration File | 85
- Sample storage-nodes.yml Configuration File | 86
- Sample k8s-host-nodes.yml Configuration File | 87
- Sample vault-data.yml Configuration File | 91

Sample site.yml Configuration File

```
# Copyright 2018 Juniper Networks, Inc. All rights reserved.
# Licensed under the Juniper Networks Script Software License (the "License").
# You may not use this script file except in compliance with the License, which is located at
# http://www.juniper.net/support/legal/scriptlicense/
# Unless required by applicable law or otherwise agreed to in writing by the parties,
# software distributed under the License is distributed on an "AS IS" BASIS,
# WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
#
#
#

global:
  # List of DNS nameservers
  dns:
    # Google Public DNS
    - "8.8.8.8"
    - "8.8.4.4"
  # List of NTP time servers
  ntp:
    # public pool.ntp.org
    - "0.pool.ntp.org"
    - "1.pool.ntp.org"
    - "2.pool.ntp.org"
    - "3.pool.ntp.org"
  # Timezone for all servers
  timezone: 'America/Los_Angeles'
  rhel:
```

```
    # Contrail Cloud Activation Key
    # These details are provided when you request an activation key from
    # contrail cloud subscriptions <contrail_cloud_subscriptions@juniper.net>
    #
    satellite:
      #SATELLITE_KEY should be defined in vault-data.yml file
      #SATELLITE_ORG
      organization: "ContrailCloud16"
      #SATELLITE_FQDN
      fqdn: contrail-cloud-satellite.juniper.net
  # DNS domain information.
  # Must be unique for every deployment to avoid name conflicts.
  # Need not be a registered DNS domain.
  domain: "my.unique.domain"

jumphost:
  network:
    # network used for provisioning (PXE booting) servers
    provision:
      # jumphost nic to be used for provisioning (PXE booting) servers
      nic: eno1

control_hosts:
  # Contains a list of label to disk mappings for roles
  disk_mapping:
    # the control host always uses the "baremetal" role
    baremetal:
      # Mapping of labels to disk devices. The label is assigned to the disk
      # device so that the disk can be referenced by the alias in other
      # configurations. for example /dev/disk/by-alias/<label>
      # Each list element contains:
      #   label: label to assign
      #   name: disk device path (e.g. /dev/sdb)
      #   OR
      #   hctl: alternative notation for disk paths specifying SCSI address
      #    (Host, Channel, Target and Lun) The HCTL can be found with the
      #    lsscsi (or lspci) command or it can be found in introspection data
      #
      - label: spinning-0
        name: /dev/sdb
      - label: spinning-1
        name: /dev/sdc
      - label: spinning-2
```

```
      name: /dev/sdd
    - label: spinning-3
      name: /dev/sde
    - label: ssd-0
      hctl: "0:2:3:0"
storage:
  # Define a set of disk groups that can be referenced for VM virtual disk allocations
  # These become virsh storage pools on the control host
  # Each pool has:
  #   mountpoint: "/absolute/path/where/lvm/will/get/mounted"
  #   type: Either "dir" or "logical".
  #       "dir" does not create any new volumes
  #         it is useful if one large hardware raid is used as /
  #       "logical" is a LVM volume placed on the list of "disk".
  #   disk: List of disk devices to use for the pool
  hdd_storage:
    mountpoint: "/srv/hdd_storage"
    type: logical
    disk:
      - "/dev/disk/by-alias/spinning-0"
      - "/dev/disk/by-alias/spinning-1"
      - "/dev/disk/by-alias/spinning-2"
      - "/dev/disk/by-alias/spinning-3"
  ssd_storage:
    mountpoint: "/srv/ssd_storage"
    type: logical
    disk:
      - "/dev/disk/by-alias/ssd-0"
  #srv:
  #  mountpoint: "/srv"
  #  type: dir

vm:
  # VM for Openstack Controller role
  control:
    disk:
      # Root disk
      vda:
        # Virsh storage pool (see storage above)
        pool: hdd_storage
  # VMs for ContrailController role
  contrail-k8s:
    disk:
```

```
        # Root disk
        vda:
          # Virsh storage pool (see storage above)
          pool: hdd_storage
    # VM for ContrailTsn role
    contrail-tsn:
      disk:
        # Root disk
        vda:
          # Virsh storage pool (see storage above)
          pool: hdd_storage
    # VM for AppFormix controller role
    appformix-controller:
      disk:
        # Root disk
        vda:
          # Virsh storage pool (see storage above)
          pool: hdd_storage

compute_hosts:
  sriov:
    #enable sriov support
    enabled: true
    #enable sriov with dpdk
    # Contrail vrouter mode:
    #   supported values are: dpdk or anything else means kernel vRouter
    mode: dpdk
    #Sriov NumVFs separated by comma
    num_vf:
      - "ens2f1:7"
    #NovaPCIPassthrough settings
    pci_passthrough:
      - devname: "ens2f1"
        physical_network: "sriov1"
  root_disk:
  # Define root disk for the listed ironic profiles.
  # The default of "/dev/sda" will be used if there is no
  # specific profile definition
  #
  # In case 'name' hint should be dropped in favor of other
  # hints the block scalar can be used:
  #
  #    ComputeKernel0Hw0: |
```

```
  #      vendor: VendorName
  #
  # which will overwrite default values.
  # To keep key-value structure and not use 'name' hint, it can be set to the
  # value which always evaluate to True, for example:
  #
  #    ComputeKernel0Hw0:
  #      name: "s!=NonExistingDevice"
  #      vendor: VendorName
  #
  # For more details please check:
  # https://docs.openstack.org/ironic/latest/install/advanced.html#specifying-the-disk-for-
deployment-root-device-hints
  #
    ComputeKernel0Hw0:
      name: "/dev/sda"
    ComputeKernel0Hw1:
      name: "/dev/sda"
    ComputeKernel1Hw1:
      name: "/dev/sda"
    ComputeKernel1Hw0:
      name: "/dev/sda"
    ComputeDpdk0Hw2:
      name: "/dev/sda"
    ComputeDpdk1Hw3:
      name: "/dev/sda"
    ComputeSriov0Hw4:
      name: "/dev/sda"
    ComputeSriov1Hw5:
      name: "/dev/sda"
  resource:
    minimal_disk:
    # This value will be used as the local_gb size for the listed ironic profiles
    # If not defined for a profile then the default will be used
      ComputeKernel0Hw0: 50
      ComputeKernel0Hw1: 50
      ComputeKernel1Hw1: 50
      ComputeKernel1Hw0: 50
      ComputeDpdk0Hw2: 50
      ComputeDpdk1Hw3: 50
      ComputeSriov0Hw4: 50
      ComputeSriov1Hw5: 50
```

```
storage_hosts:
  root_disk:
  # Define root disk for the listed ironic profiles.
  # The default of "/dev/sda" will be used if there is no
  # specific profile definition
    CephStorage0Hw6:
      name: "/dev/sda"
    CephStorage1Hw7:
      name: "/dev/sda"

undercloud:
  nova:
    # Nova flavor definitions for roles
    flavor:
      CephStorage0Hw6:
        cpu: 1
        memory: 4
        disk: 40
        ephemeral: 0
      CephStorage1Hw7:
        cpu: 1
        memory: 4
        disk: 40
        ephemeral: 0
      ComputeKernel0Hw0:
        cpu: 8
        memory: 24
        disk: 40
        ephemeral: 0
      ComputeKernel0Hw1:
        cpu: 8
        memory: 24
        disk: 40
        ephemeral: 0
      ComputeKernel1Hw1:
        cpu: 8
        memory: 24
        disk: 40
        ephemeral: 0
      ComputeKernel1Hw0:
        cpu: 8
        memory: 24
        disk: 40
```

```
        ephemeral: 0
      ComputeDpdk0Hw2:
        cpu: 8
        memory: 24
        disk: 40
        ephemeral: 0
      ComputeDpdk1Hw3:
        cpu: 8
        memory: 24
        disk: 40
        ephemeral: 0
      ComputeSriov0Hw4:
        cpu: 8
        memory: 24
        disk: 40
        ephemeral: 0
      ComputeSriov1Hw5:
        cpu: 8
        memory: 24
        disk: 40
        ephemeral: 0

k8s:
  external_vip_ip: 10.10.10.102
  internal_api_vip_ip: 172.16.0.91

overcloud:
  # Contains a list of label to disk mappings for roles.
  # When Ceph Storage is disabled, compute-related roles (Compute* and
  # ComputeDpdk* roles) will use any disks labeled with
  # "ephemeral-<digits>" for local Nova ephemeral storage.
  disk_mapping:
    ComputeKernel:
      # Mapping of labels to disk devices. The label is assigned to the disk
      # device so that the disk can be referenced by the alias in other
      # configurations. for example /dev/disk/by-alias/<label>
      # Each list element contains:
      #   label: label to assign
      #   hctl: disk device path H:C:T:L (the path must exist). see lsscsi
      - label: ephemeral-0
        hctl: '5:0:0:0'
      - label: ephemeral-1
        hctl: '6:0:0:0'
```

```
    - label: ephemeral-2
      hctl: '7:0:0:0'
    - label: ephemeral-3
      hctl: '8:0:0:0'
  ComputeKernel0Hw0:
    # Mapping of labels to disk devices. The label is assigned to the disk
    # device so that the disk can be referenced by the alias in other
    # configurations. for example /dev/disk/by-alias/<label>
    # Each list element contains:
    #   label: label to assign
    #   hctl: disk device path H:C:T:L (the path must exist). see lsscsi
    - label: ephemeral-0
      hctl: '5:0:0:0'
    - label: ephemeral-1
      hctl: '6:0:0:0'
    - label: ephemeral-2
      hctl: '7:0:0:0'
    - label: ephemeral-3
      hctl: '8:0:0:0'
  ComputeKernel1Hw0:
    # Mapping of labels to disk devices. The label is assigned to the disk
    # device so that the disk can be referenced by the alias in other
    # configurations. for example /dev/disk/by-alias/<label>
    # Each list element contains:
    #   label: label to assign
    #   hctl: disk device path H:C:T:L (the path must exist). see lsscsi
    - label: ephemeral-0
      hctl: '5:0:0:0'
    - label: ephemeral-1
      hctl: '6:0:0:0'
    - label: ephemeral-2
      hctl: '7:0:0:0'
    - label: ephemeral-3
      hctl: '8:0:0:0'
  ComputeKernel1Hw1:
    # Mapping of labels to disk devices. The label is assigned to the disk
    # device so that the disk can be referenced by the alias in other
    # configurations. for example /dev/disk/by-alias/<label>
    # Each list element contains:
    #   label: label to assign
    #   hctl: disk device path H:C:T:L (the path must exist). see lsscsi
    - label: ephemeral-0
      hctl: '5:0:0:0'
```

```
    - label: ephemeral-1
      hctl: '6:0:0:0'
    - label: ephemeral-2
      hctl: '7:0:0:0'
    - label: ephemeral-3
      hctl: '8:0:0:0'
  ComputeKernel0Hw1:
    # Mapping of labels to disk devices. The label is assigned to the disk
    # device so that the disk can be referenced by the alias in other
    # configurations. for example /dev/disk/by-alias/<label>
    # Each list element contains:
    #   label: label to assign
    #   hctl: disk device path H:C:T:L (the path must exist). see lsscsi
    - label: ephemeral-0
      hctl: '5:0:0:0'
    - label: ephemeral-1
      hctl: '6:0:0:0'
    - label: ephemeral-2
      hctl: '7:0:0:0'
    - label: ephemeral-3
      hctl: '8:0:0:0'
  ComputeDpdk:
    # Mapping of labels to disk devices. The label is assigned to the disk
    # device so that the disk can be referenced by the alias in other
    # configurations. for example /dev/disk/by-alias/<label>
    # Each list element contains:
    #   label: label to assign
    #   hctl: disk device path H:C:T:L (the path must exist). see lsscsi
    - label: ephemeral-0
      hctl: '5:0:0:0'
    - label: ephemeral-1
      hctl: '6:0:0:0'
    - label: ephemeral-2
      hctl: '7:0:0:0'
    - label: ephemeral-3
      hctl: '8:0:0:0'
  ComputeDpdk0Hw2:
    # Mapping of labels to disk devices. The label is assigned to the disk
    # device so that the disk can be referenced by the alias in other
    # configurations. for example /dev/disk/by-alias/<label>
    # Each list element contains:
    #   label: label to assign
    #   hctl: disk device path H:C:T:L (the path must exist). see lsscsi
```

```
    - label: ephemeral-0
      hctl: '5:0:0:0'
    - label: ephemeral-1
      hctl: '6:0:0:0'
    - label: ephemeral-2
      hctl: '7:0:0:0'
    - label: ephemeral-3
      hctl: '8:0:0:0'
  ComputeDpdk1Hw3:
    # Mapping of labels to disk devices. The label is assigned to the disk
    # device so that the disk can be referenced by the alias in other
    # configurations. for example /dev/disk/by-alias/<label>
    # Each list element contains:
    #   label: label to assign
    #   hctl: disk device path H:C:T:L (the path must exist). see lsscsi
    - label: ephemeral-0
      hctl: '5:0:0:0'
    - label: ephemeral-1
      hctl: '6:0:0:0'
    - label: ephemeral-2
      hctl: '7:0:0:0'
    - label: ephemeral-3
      hctl: '8:0:0:0'
  ComputeSriov:
    # Mapping of labels to disk devices. The label is assigned to the disk
    # device so that the disk can be referenced by the alias in other
    # configurations. for example /dev/disk/by-alias/<label>
    # Each list element contains:
    #   label: label to assign
    #   hctl: disk device path H:C:T:L (the path must exist). see lsscsi
    - label: ephemeral-0
      hctl: '5:0:0:0'
    - label: ephemeral-1
      hctl: '6:0:0:0'
    - label: ephemeral-2
      hctl: '7:0:0:0'
    - label: ephemeral-3
      hctl: '8:0:0:0'
  ComputeSriov0Hw4:
    # Mapping of labels to disk devices. The label is assigned to the disk
    # device so that the disk can be referenced by the alias in other
    # configurations. for example /dev/disk/by-alias/<label>
    # Each list element contains:
```

```
      #   label: label to assign
      #   hctl: disk device path H:C:T:L (the path must exist). see lsscsi
      - label: ephemeral-0
        hctl: '5:0:0:0'
      - label: ephemeral-1
        hctl: '6:0:0:0'
      - label: ephemeral-2
        hctl: '7:0:0:0'
      - label: ephemeral-3
        hctl: '8:0:0:0'
    ComputeSriov1Hw5:
      # Mapping of labels to disk devices. The label is assigned to the disk
      # device so that the disk can be referenced by the alias in other
      # configurations. for example /dev/disk/by-alias/<label>
      # Each list element contains:
      #   label: label to assign
      #   hctl: disk device path H:C:T:L (the path must exist). see lsscsi
      - label: ephemeral-0
        hctl: '5:0:0:0'
      - label: ephemeral-1
        hctl: '6:0:0:0'
      - label: ephemeral-2
        hctl: '7:0:0:0'
      - label: ephemeral-3
        hctl: '8:0:0:0'
  extra_config:
    ComputeDpdkParameters:
      TunedProfileName: "cpu-partitioning"
      ContrailDpdkOptions: "--vr_flow_entries=2000000 --yield_option 0"
      KernelArgs: "intel_iommu=on iommu=pt default_hugepagesz=1GB hugepagesz=1G hugepages=64
hugepagesz=2M hugepages=2048 isolcpus=2-9,22-29"
      IsolCpusList: "2-9,22-29"
      NovaVcpuPinSet: ['4-9','24-29']
      ContrailSettings:
        SERVICE_CORE_MASK: '0x1'
        DPDK_CTRL_THREAD_MASK: '0x1'
      ExtraSysctlSettings:
        vm.nr_hugepages:
          value: 64
        vm.max_map_count:
          value: 128960
    ComputeDpdk0Hw2Parameters:
      TunedProfileName: "cpu-partitioning"
```

```
      ContrailDpdkOptions: "--vr_flow_entries=2000000 --yield_option 0"
      KernelArgs: "intel_iommu=on iommu=pt default_hugepagesz=1GB hugepagesz=1G hugepages=64
hugepagesz=2M hugepages=2048 isolcpus=2-9,22-29"
      IsolCpusList: "2-9,22-29"
      NovaVcpuPinSet: ['4-9','24-29']
      ContrailSettings:
        SERVICE_CORE_MASK: '0x1'
        DPDK_CTRL_THREAD_MASK: '0x1'
      ExtraSysctlSettings:
        vm.nr_hugepages:
          value: 64
        vm.max_map_count:
          value: 128960
    ComputeDpdk1Hw3Parameters:
      TunedProfileName: "cpu-partitioning"
      ContrailDpdkOptions: "--vr_flow_entries=2000000 --yield_option 0"
      KernelArgs: "intel_iommu=on iommu=pt default_hugepagesz=1GB hugepagesz=1G hugepages=64
hugepagesz=2M hugepages=2048 isolcpus=2-9,22-29"
      IsolCpusList: "2-9,22-29"
      NovaVcpuPinSet: ['4-9','24-29']
      ContrailSettings:
        SERVICE_CORE_MASK: '0x1'
        DPDK_CTRL_THREAD_MASK: '0x1'
      ExtraSysctlSettings:
        vm.nr_hugepages:
          value: 64
        vm.max_map_count:
          value: 128960
    ComputeKernelParameters:
      ContrailVrouterHugepages1GB: 64
      ContrailVrouterHugepages2MB: 2048
      KernelArgs: "intel_iommu=on iommu=pt default_hugepagesz=1GB hugepagesz=1G hugepages=64
hugepagesz=2M hugepages=2048"
      ExtraSysctlSettings:
        vm.nr_hugepages:
          value: 64
        vm.max_map_count:
          value: 128960
    ComputeKernel0Hw0Parameters:
      ContrailVrouterHugepages1GB: 64
      ContrailVrouterHugepages2MB: 2048
      KernelArgs: "intel_iommu=on iommu=pt default_hugepagesz=1GB hugepagesz=1G hugepages=64
hugepagesz=2M hugepages=2048"
```

```
      ExtraSysctlSettings:
        vm.nr_hugepages:
          value: 64
        vm.max_map_count:
          value: 128960
    ComputeKernel0Hw1Parameters:
      ContrailVrouterHugepages1GB: 64
      ContrailVrouterHugepages2MB: 2048
      KernelArgs: "intel_iommu=on iommu=pt default_hugepagesz=1GB hugepagesz=1G hugepages=64
hugepagesz=2M hugepages=2048"
      ExtraSysctlSettings:
        vm.nr_hugepages:
          value: 64
        vm.max_map_count:
          value: 128960
    ComputeKernel1Hw0Parameters:
      ContrailVrouterHugepages1GB: 64
      ContrailVrouterHugepages2MB: 2048
      KernelArgs: "intel_iommu=on iommu=pt default_hugepagesz=1GB hugepagesz=1G hugepages=64
hugepagesz=2M hugepages=2048"
      ExtraSysctlSettings:
        vm.nr_hugepages:
          value: 64
        vm.max_map_count:
          value: 128960
    ComputeKernel1Hw1Parameters:
      ContrailVrouterHugepages1GB: 64
      ContrailVrouterHugepages2MB: 2048
      KernelArgs: "intel_iommu=on iommu=pt default_hugepagesz=1GB hugepagesz=1G hugepages=64
hugepagesz=2M hugepages=2048"
      ExtraSysctlSettings:
        vm.nr_hugepages:
          value: 64
        vm.max_map_count:
          value: 128960

  network:
    # The external network is used for referencing the overcloud APIs from outside the
infrastructure.
    external:
      # Network name used by TripleO Heat Templates
      heat_name: External
      # CIDR (IP/prefix) for the external network subnet
```

```
    # Corresponds to the ExternalIpSubnet heat property
    cidr: "10.84.36.64/28"
    # Default route for the external network
    # Corresponds to the ExternalInterfaceDefaultRoute heat property
    gateway: "10.84.36.78"
    # VLAN tag for the external network
    # Corresponds to the ExternalNetworkVlanID heat property
    vlan: 1350
    # Floating virtual IP for the Openstack APIs on the external network
    # Corresponds to the PublicVirtualFixedIPs heat property
    vip: "10.84.36.77"
    # DHCP pool for the external network
    # Be sure that the range is large enough to accomodate all nodes in the external network
    pool:
      # Range start for the DHCP pool
      start: "10.84.36.65"
      # Range end for the DHCP pool
      end: "10.84.36.75"
    # MTU for external network
    # Corresponds to the ExternalNetworkMtu heat property
    mtu: 9000
    # List of roles that can be on this network
    role:
      - Controller
      - AppformixController
  # The internal API network is used for control plane signalling and service API calls
  internal_api:
    # Network name used by TripleO Heat Templates
    heat_name: InternalApi
    # VLAN tag for the internal API network
    # Corresponds to the InternalApiNetworkVlanID heat property
    vlan: 1200
    # CIDR (IP/prefix) for the internal api supernet network subnet
    # Corresponds to the InternalApiSupernet heat property
    # Supernet is used in spine/leaf configuration
    # Supernet accommodate all related leaf networks, e.g. internal_api0 and internal_api1
    # Supernet is used to create static routes between leafs
    # Supernet is defined only for main network, not per leafs
    supernet: "172.16.0.0/22"
    # CIDR (IP/prefix) for the internal api network subnet
    # Corresponds to the InternalApiIpSubnet heat property
    cidr: "172.16.0.0/24"
    # Default route for the internal api network
```

```
      # Corresponds to the InternalApiInterfaceDefaultRoute heat property
      gateway: 172.16.0.1
      # MTU for internal api network
      # Corresponds to the InternalApiNetworkMtu heat property
      mtu: 9000
      # DHCP pool for the internal api network
      # Be sure that the range is large enough to accomodate all nodes in the internal api
network
      pool:
        # Range start for the DHCP pool
        start: 172.16.0.100
        # Range end for the DHCP pool
        end: 172.16.0.160
      # Floating virtual IP for the Openstack APIs on the internal api network
      # Corresponds to the InternalApiVirtualFixedIPs heat property
      vip: 172.16.0.90
      # List of roles that can be on this network
      role:
        - Controller
        - ContrailTsn
        - AppformixController
        - ComputeKernel
        - ComputeDpdk
        - ComputeSriov
    # Leaf 0 subnet of the internal_api network
    internal_api0:
      # Network name used by TripleO Heat Templates
      heat_name: InternalApi0
      # VLAN tag for the internal API 0 network
      # Corresponds to the InternalApi0NetworkVlanID heat property
      vlan: 1201
      # CIDR (IP/prefix) for the internal api 0 network subnet
      # Corresponds to the InternalApi0IpSubnet heat property
      cidr: "172.16.1.0/24"
      # Default route for the internal api 0 network
      # Corresponds to the InternalApi0InterfaceDefaultRoute heat property
      gateway: 172.16.1.1
      # MTU for internal api 0 network
      # Corresponds to the InternalApi0NetworkMtu heat property
      mtu: 9000
      # DHCP pool for the internal api 0 network
      # Be sure that the range is large enough to accomodate all nodes in the internal api
network
```

```
      pool:
        # Range start for the DHCP pool
        start: 172.16.1.100
        # Range end for the DHCP pool
        end: 172.16.1.200
      # List of roles that can be on this network
      role:
        - ComputeKernel0Hw0
        - ComputeKernel0Hw1
        - ComputeSriov0Hw4
        - ComputeDpdk0Hw2
    # Leaf 1 subnet of the internal_api network
    internal_api1:
      # Network name used by TripleO Heat Templates
      heat_name: InternalApi1
      # VLAN tag for the internal API 1 network
      # Corresponds to the InternalApi1NetworkVlanID heat property
      vlan: 1202
      # CIDR (IP/prefix) for the internal api 1 network subnet
      # Corresponds to the InternalApi1IpSubnet heat property
      cidr: "172.16.2.0/24"
      # Default route for the internal api 1 network
      # Corresponds to the InternalApi1InterfaceDefaultRoute heat property
      gateway: 172.16.2.1
      # MTU for internal api 1 network
      # Corresponds to the InternalApi1NetworkMtu heat property
      mtu: 9000
      # DHCP pool for the internal api 1 network
      # Be sure that the range is large enough to accomodate all nodes in the internal api
network
      pool:
        # Range start for the DHCP pool
        start: 172.16.2.100
        # Range end for the DHCP pool
        end: 172.16.2.200
      # List of roles that can be on this network
      role:
        - ComputeSriov1Hw5
        - ComputeKernel1Hw0
        - ComputeDpdk1Hw3
        - ComputeKernel1Hw1
    # The management network is defined for backwards-compatibility in RHOSP and is not
    # used by default by any roles.
```

```
  management:
    # Network name used by TripleO Heat Templates
    heat_name: Management
    # VLAN tag for the management network
    # Corresponds to the ManagementNetworkVlanID heat property
    vlan: 1300
    # CIDR (IP/prefix) for the network subnet
    # Corresponds to the ManagementIpSubnet heat property
    cidr: "192.168.1.0/24"
    # MTU for the network
    # Corresponds to the ManagementNetworkMtu heat property
    mtu: 9000
    # DHCP pool for the  network
    # Be sure that the range is large enough to accomodate all nodes in the network
    pool:
      # Range start for the DHCP pool
      start: 192.168.1.100
      # Range end for the DHCP pool
      end: 192.168.1.200
  # The storage network is used for Compute storage access
  storage:
    # Network name used by TripleO Heat Templates
    heat_name: Storage
    # VLAN tag for the storage network
    # Corresponds to the StorageNetworkVlanID heat property
    vlan: 1500
    supernet: "172.16.16.0/22"
    cidr: "172.16.16.0/24"
    gateway: 172.16.16.1
    mtu: 9000
    pool:
      start: 172.16.16.100
      end: 172.16.16.200
    # List of roles that can be on this network
    role:
      - Controller
      - CephStorage
      - ComputeKernel
      - ComputeDpdk
      - ComputeSriov
      - ContrailTsn
  # Leaf 0 subnet of the storage network
  storage0:
```

```
    # Network name used by TripleO Heat Templates
    heat_name: Storage0
    vlan: 1501
    cidr: "172.16.17.0/24"
    gateway: 172.16.17.1
    mtu: 9000
    pool:
      start: 172.16.17.100
      end: 172.16.17.200
    # List of roles that can be on this network
    role:
      - CephStorage0Hw6
      - ComputeKernel0Hw0
      - ComputeKernel0Hw1
      - ComputeDpdk0Hw2
      - ComputeSriov0Hw4
  # Leaf 1 subnet of the storage network
  storage1:
    # Network name used by TripleO Heat Templates
    heat_name: Storage1
    vlan: 1502
    cidr: "172.16.18.0/24"
    gateway: 172.16.18.1
    mtu: 9000
    pool:
      start: 172.16.18.100
      end: 172.16.18.200
    # List of roles that can be on this network
    role:
      - ComputeSriov1Hw5
      - ComputeKernel1Hw0
      - ComputeDpdk1Hw3
      - ComputeKernel1Hw1
      - CephStorage1Hw7
  # The storage management network is used for storage operations such as replication
  storage_mgmt:
    # Network name used by TripleO Heat Templates
    heat_name: StorageMgmt
    # VLAN tag for the storage management network
    # Corresponds to the StorageMgmtNetworkVlanID heat property
    vlan: 1450
    supernet: "172.16.20.0/22"
    cidr: "172.16.20.0/24"
```

```
    gateway: 172.16.20.1
    vip_enable: false
    mtu: 9000
    pool:
      start: 172.16.20.100
      end: 172.16.20.200
    # List of roles that can be on this network
    role:
      - Controller
  # Leaf 0 subnet of the storage_mgmt network
  storage_mgmt0:
    # Network name used by TripleO Heat Templates
    heat_name: StorageMgmt0
    vlan: 1451
    cidr: "172.16.21.0/24"
    gateway: 172.16.21.1
    mtu: 9000
    pool:
      start: 172.16.21.100
      end: 172.16.21.200
    # List of roles that can be on this network
    role:
      - CephStorage0Hw6
  # Leaf 1 subnet of the storage_mgmt network
  storage_mgmt1:
    # Network name used by TripleO Heat Templates
    heat_name: StorageMgmt1
    vlan: 1452
    cidr: "172.16.22.0/24"
    gateway: 172.16.22.1
    mtu: 9000
    pool:
      start: 172.16.22.100
      end: 172.16.22.200
    # List of roles that can be on this network
    role:
      - CephStorage1Hw7
  # The tenant network is used for tenant workload data
  tenant:
    # Network name used by TripleO Heat Templates
    heat_name: Tenant
    # VLAN tag for the tenant network
    # Corresponds to the TenantNetworkVlanID heat property
```

```
    vlan: 1250
    supernet: "172.16.80.0/21"
    cidr: "172.16.81.0/24"
    gateway: 172.16.81.1
    vrouter_gateway: 172.16.81.1
    mtu: 9000
    pool:
      start: 172.16.81.100
      end: 172.16.81.200
    # List of roles that can be on this network
    role:
      - ContrailTsn
      - ComputeKernel
      - ComputeDpdk
      - ComputeSriov
  # Leaf 0 subnet of the tenant network
  tenant0:
    # Network name used by TripleO Heat Templates
    heat_name: Tenant0
    vlan: 1251
    cidr: "172.16.82.0/24"
    gateway: 172.16.82.1
    vrouter_gateway: 172.16.82.1
    mtu: 9000
    pool:
      start: 172.16.82.100
      end: 172.16.82.200
    # List of roles that can be on this network
    role:
      - ComputeKernel0Hw0
      - ComputeKernel0Hw1
      - ComputeDpdk0Hw2
      - ComputeSriov0Hw4
  # Leaf 1 subnet of the tenant network
  tenant1:
    # Network name used by TripleO Heat Templates
    heat_name: Tenant1
    vlan: 1252
    cidr: "172.16.83.0/24"
    gateway: 172.16.83.1
    vrouter_gateway: 172.16.83.1
    mtu: 9000
    pool:
```

```
        start: 172.16.83.100
        end: 172.16.83.200
      # List of roles that can be on this network
      role:
        - ComputeSriov1Hw5
        - ComputeKernel1Hw0
        - ComputeDpdk1Hw3
        - ComputeKernel1Hw1
  # Contrail sepcific settings
  #contrail:
  #  aaa_mode: cloud-admin
  #  vrouter:
  #    contrail_settings:
  #      # Settings per profile.
  #      # Profile's contrail_settings replace default settings and should include
  #      # all keys and values which are intended to be exported on given role.
  #      # When leafs are used it implies per profile configuration as it defines
  #      # VROUTER_GATEWAY for profile by quering node's tenant network for
  #      # vrouter_gateway value.
  #      default:
  #        VROUTER_GATEWAY: 172.16.81.254
  #        BGP_ASN: 64512
  #        LACP_RATE: 1
  #      ComputeKernel1Hw0:
  #        LACP_RATE: 1

  # Information used to generate the SSL certificates for the public Openstack service APIs
  tls:
    #countryName_default
    country: "US"
    #stateOrProvinceName_default
    state: "CA"
    #localityName_default
    city: "Sunnyvale"
    #organizationalUnitName_default
    organization: "JNPR"
    #commonName_default - this is typically the external VIP
    common_name: "10.10.10.90"

ceph:
  # Choice to enable Ceph storage in the overcloud.
  # "true" means that Ceph will be deployed as the backed for Cinder and Glance services.
  # "false" false means that Ceph will not be deployed.
```

```
  enabled: true
  # Ceph OSD disk configuration
  osd:
    # Update the Ceph crush map when OSDs are started
    crush_update_on_start: true
    # Ceph OSD disk assignments. The named disks will be exclusively used by Ceph for
persistence.
    # Lvm is a default scenario for ceph deployment with bluestore as a backend.
    # When all named disks are the same type, spinning or solid state, all of them will be used
    # as ceph osds. When disks with mixed types are defined spinning disks will be used as osds
    # and on solid state disks ceph db will be created. For mixed types of disks the automatic
pgp
    # number calculation requires assigning key 'contents' with value 'db' to ssd disks.
    # In below example disks sd[b-e] are spinning disks and sdf is solid state disk.
    default:
      disk:
        '/dev/sdb':
        '/dev/sdc':
        '/dev/sdd':
        '/dev/sde':
        '/dev/sdf':
          contents: db
    CephStorage0Hw6:
      disk:
        '/dev/sdb':
        '/dev/sdc':
        '/dev/sdd':
        '/dev/sde':
        '/dev/sdf':
          contents: db
    CephStorage1Hw7:
      disk:
        '/dev/sdb':
        '/dev/sdc':
        '/dev/sdd':
        '/dev/sde':
        '/dev/sdf':
          contents: db
  # By default, pgp number is calculated by contrail cloud. If you want, you can give this
parameter
  # by yourself. Use the calculator on the website: https://ceph.com/pgcalc/. Calculator takes
into
  # account also pool utilization. Calculated pgp_num can be introduced in configuration as
```

```
below.
  # It's defined per used pool.
  #  pool:
  #    vms:
  #      pgp_num: 32
  #    rbd:
  #      pgp_num: 32
  #    images:
  #      pgp_num: 32
  #    volumes:
  #      pgp_num: 32
  #    backups:
  #      pgp_num: 32
  #
  # Rados Gateway when enabled, which is a default behaviour, creates it's own ceph pools
  # not tracked by contrail cloud. Those pools can be predefined to better control
  # their sizes. Below pools definitions are not an exhaustive, please consult with
  # https://ceph.com/pgcalc/
  # Pools should have enabled application according to their use.
  # If not changed explicit, pools are created with 'rbd' application assigned.
  # Available options are:
  #   - rbd for the Ceph Block Device
  #   - rgw for the Ceph Object Gateway
  #   - cephfs for the Ceph Filesystem
  # or user defined value for custom application.
  # More details can be found on
  # https://access.redhat.com/documentation/en-us/red_hat_ceph_storage/3/html/
storage_strategies_guide/pools-1#enable-application
  #    .rgw.root:
  #      pgp_num: 16
  #      enabled: true
  #      replica: 3
  #      application: rgw
  #    default.rgw.control:
  #      pgp_num: 16
  #      enabled: true
  #      replica: 3
  #      application: rgw
  #    default.rgw.meta:
  #      pgp_num: 16
  #      enabled: true
  #      replica: 3
  #      application: rgw
```

```
  #    default.rgw.log:
  #      pgp_num: 16
  #      enabled: true
  #      replica: 3
  #      application: rgw
  #    default.rgw.buckets.index:
  #      pgp_num: 16
  #      enabled: true
  #      replica: 3
  #      application: rgw
  #    default.rgw.buckets.data:
  #      pgp_num: 16
  #      enabled: true
  #      replica: 3
  #      application: rgw
  #    default.rgw.buckets.non-ec:
  #      pgp_num: 16
  #      enabled: true
  #      replica: 3
  #      application: rgw

appformix:
  # Set to true if you have multiple control hosts which allows Apformix to run in HA mode
  enable_ha: true
  # Floating virtual IP for the Appformix APIs on the external network, used and required by HA
mode.
  vip: "10.10.10.101"
  # Floating virtual IP for the Appformix APIs on the internal network, used and required by HA
mode.
  secondary_vip: "172.16.0.89"
  keepalived:
    # Set which interface will be used for vrrp
    vrrp_interface: "enp2s0"
    # Set which interface will be used for second vrrp
    secondary_vrrp_interface: "vlan1200"
```

Sample inventory.yml Configuration File

```
# Copyright 2018 Juniper Networks, Inc. All rights reserved.
# Licensed under the Juniper Networks Script Software License (the "License").
# You may not use this script file except in compliance with the License, which is located at
# http://www.juniper.net/support/legal/scriptlicense/
# Unless required by applicable law or otherwise agreed to in writing by the parties,
# software distributed under the License is distributed on an "AS IS" BASIS,
# WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
#
#
#

# Common values shared among group of nodes
ipmi_hardware1: &hardware1
  pm_type: "ipmi"
  pm_user: "{{ vault['inventory_nodes']['hardware1']['pm_user'] }}"
  pm_password: "{{ vault['inventory_nodes']['hardware1']['pm_password'] }}"
  capabilities: "boot_mode:uefi"

# List of baremetal server nodes that can be used for the deploying roles
# Each list item contains:
#    name: logical name to assign this resource (string)
#    pm_addr: IP address for resourceIPMI interface (string)
#    pm_type: Ironic driver to interface with this resource (typically ipmi) (string)
#    pm_user: IPMI user account (string)
#    pm_password: IPMI account user password (string)
#    capabilities: String of comma separated list of node capabilities.
#                  Capabilities 'profile' and 'boot_option' are managed
#                  by Contrail Cloud and will be omitted. (string)
#                  e.g capabilities: "boot_mode:uefi" set boot mode to uefi
#
# Some values common for nodes can be moved to dedicated section like ipmi_hardware1
# and be referred like this:
#    <<: *hardware1
inventory_nodes:
  - name: "control-host1"
    pm_addr: "10.10.11.58"
    <<: *hardware1
  - name: "control-host2"
    pm_addr: "10.10.11.59"
```

```
    <<: *hardware1
  - name: "control-host3"
    pm_addr: "10.10.11.60"
    <<: *hardware1
  - name: "storage1"
    pm_addr: "10.10.11.61"
    <<: *hardware1
  - name: "storage2"
    pm_addr: "10.10.11.62"
    <<: *hardware1
  - name: "storage3"
    pm_addr: "10.10.11.63"
    <<: *hardware1
  - name: "computedpdk1"
    pm_addr: "10.10.11.64"
    <<: *hardware1
  - name: "computedpdk2"
    pm_addr: "10.10.11.65"
    <<: *hardware1
  - name: "compute1"
    pm_addr: "10.10.11.66"
    <<: *hardware1
  - name: "compute2"
    pm_addr: "10.10.11.67"
    <<: *hardware1
  - name: "compute3"
    pm_addr: "10.10.11.68"
    <<: *hardware1
  - name: "compute4"
    pm_addr: "10.10.11.69"
    <<: *hardware1
  - name: "computesriov1"
    pm_addr: "10.10.11.70"
    <<: *hardware1
  - name: "computesriov2"
    pm_addr: "10.10.11.71"
    <<: *hardware1
```

Sample control-host-nodes.yml Configuration File

```
# Copyright 2020 Juniper Networks, Inc. All rights reserved.
# Licensed under the Juniper Networks Script Software License (the "License").
# You may not use this script file except in compliance with the License, which is located at
# http://www.juniper.net/support/legal/scriptlicense/
# Unless required by applicable law or otherwise agreed to in writing by the parties,
# software distributed under the License is distributed on an "AS IS" BASIS,
# WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
#
#
#

# List of nodes to use as control host role
# Each list item contains a set of variables which can be referenced
# with "{{ host.<variable> }}" in control_host_nodes_network_config below.
# Other ad-hoc variables can be added as needed.
#    name: name of a node in the inventory (string)
#    hostname: hostname to assign the node after it is imaged (string)
#    control_ip_netmask: static CIDR address on Control Plane network.
#                        Choose a value outside the DHCP range. (string)
#    dns_server1,dns_server2: dns server addresses (string)
#    max_mtu: The largest MTU supported by an interface
#

control_host_nodes:
  - name: "control-host1"
    control_ip_netmask: "192.168.213.5/24"
    max_mtu: 9216
  - name: "control-host2"
    control_ip_netmask: "192.168.213.6/24"
    max_mtu: 9216
  - name: "control-host3"
    control_ip_netmask: "192.168.213.7/24"
    max_mtu: 9216

# Template for network layout on all control host nodes
# This follows the os-net-config syntax
# See https://github.com/openstack/os-net-config/tree/stable/queens
# variables from control_host_nodes can be refered with "{{ host.<variable> }}"
```

```
control_host_nodes_network_config:
  - type: interface
    name: eno1
    use_dhcp: true
    mtu: "{{ host.max_mtu }}"
    nm_controlled: true
  - type: interface
    name: eno2
    use_dhcp: false
    mtu: "{{ host.max_mtu }}"
    nm_controlled: true
  - type: linux_bond
    name: bond0
    use_dhcp: false
    mtu: "{{ host.max_mtu }}"
    bonding_options: "mode=802.3ad xmit_hash_policy=layer3+4 lacp_rate=fast miimon=100"
    nm_controlled: true
    members:
      -
        type: interface
        name: ens7f0
        use_dhcp: false
        mtu: "{{ host.max_mtu }}"
        primary: true
        nm_controlled: true
      -
        type: interface
        name: ens7f1
        use_dhcp: false
        mtu: "{{ host.max_mtu }}"
        nm_controlled: true

control_hosts:
  # The mapping from control host interfaces to the control VM interfaces
  # The first interface (enp1s0) must always be the Control Plane network to allow the VM to PXE
boot
  # VM interface names must be sequential with no gaps (e.g. enp1s0, enp2s0, enp3s0,...)
  vm_interfaces:
    - interface: enp1s0
      physical_interface: eno1
    - interface: enp2s0
      physical_interface: eno2
```

```
    - interface: enp3s0
      physical_interface: bond0
```

Sample overcloud-nics.yml Configuration File

```
Controller_network_config:
  - type: interface
    name: enp1s0
    dns_servers:
      get_param: DnsServers
    use_dhcp: false
    mtu:
      get_param: ControlPlaneMtu
    addresses:
    - ip_netmask:
        list_join:
          - '/'
          - - get_param: ControlPlaneIp
            - get_param: ControlPlaneSubnetCidr
  - type: vlan
    device: enp1s0
    vlan_id:
      get_param: StorageNetworkVlanID
    mtu:
      get_param: StorageMtu
    addresses:
    - ip_netmask:
        get_param: StorageIpSubnet
    routes:
    -
      ip_netmask:
        get_param: StorageSupernet
      next_hop:
        get_param: StorageInterfaceDefaultRoute
  - type: vlan
    device: enp1s0
    vlan_id:
      get_param: StorageMgmtNetworkVlanID
    mtu:
```

```
      get_param: StorageMgmtMtu
    addresses:
    - ip_netmask:
        get_param: StorageMgmtIpSubnet
    routes:
    -
      ip_netmask:
        get_param: StorageMgmtSupernet
      next_hop:
        get_param: StorageMgmtInterfaceDefaultRoute
  - type: vlan
    device: enp1s0
    vlan_id:
      get_param: InternalApiNetworkVlanID
    mtu:
      get_param: InternalApiMtu
    addresses:
    - ip_netmask:
        get_param: InternalApiIpSubnet
    routes:
    -
      ip_netmask:
        get_param: InternalApiSupernet
      next_hop:
        get_param: InternalApiInterfaceDefaultRoute
  - type: interface
    name: enp2s0
    mtu:
      get_param: ExternalMtu
    addresses:
    - ip_netmask:
        get_param: ExternalIpSubnet
    routes:
    -
      default: True
      next_hop:
        get_param: ExternalInterfaceDefaultRoute
  - type: interface
    name: enp3s0
    use_dhcp: false

AppformixController_network_config:
  - type: interface
```

```
    name: enp1s0
    dns_servers:
      get_param: DnsServers
    use_dhcp: false
    mtu:
      get_param: ControlPlaneMtu
    addresses:
    - ip_netmask:
        list_join:
          - '/'
          - - get_param: ControlPlaneIp
            - get_param: ControlPlaneSubnetCidr
  - type: vlan
    device: enp1s0
    vlan_id:
      get_param: InternalApiNetworkVlanID
    mtu:
      get_param: InternalApiMtu
    addresses:
    - ip_netmask:
        get_param: InternalApiIpSubnet
    routes:
    -
      ip_netmask:
        get_param: InternalApiSupernet
      next_hop:
        get_param: InternalApiInterfaceDefaultRoute
  - type: interface
    name: enp2s0
    mtu:
      get_param: ExternalMtu
    addresses:
    - ip_netmask:
        get_param: ExternalIpSubnet
    routes:
    -
      default: True
      next_hop:
        get_param: ExternalInterfaceDefaultRoute
  - type: interface
    name: enp3s0
    use_dhcp: false
```

```
ContrailTsn_network_config:
  - type: interface
    name: enp1s0
    dns_servers:
      get_param: DnsServers
    mtu:
      get_param: ControlPlaneMtu
    addresses:
    - ip_netmask:
        list_join:
          - '/'
          - - get_param: ControlPlaneIp
            - get_param: ControlPlaneSubnetCidr
    use_dhcp: false
    routes:
    -
      default: True
      next_hop:
        get_param: ControlPlaneDefaultRoute
  - type: vlan
    device: enp1s0
    vlan_id:
      get_param: InternalApiNetworkVlanID
    mtu:
      get_param: InternalApiMtu
    addresses:
    - ip_netmask:
        get_param: InternalApiIpSubnet
    routes:
    -
      ip_netmask:
        get_param: InternalApiSupernet
      next_hop:
        get_param: InternalApiInterfaceDefaultRoute
  - type: interface
    name: enp2s0
    use_dhcp: false
  - type: interface
    name: enp3s0
    use_dhcp: false
    mtu:
      get_param: TenantMtu
  - type: vlan
```

```
    device: enp3s0
    vlan_id:
      get_param: TenantNetworkVlanID
    mtu:
      get_param: TenantMtu
    use_dhcp: false
  - type: contrail_vrouter
    name: vhost0
    members:
      -
        type: interface
        name:
          str_replace:
            template: vlanVLANID
            params:
              VLANID: {get_param: TenantNetworkVlanID}
        use_dhcp: false
    mtu:
      get_param: TenantMtu
    addresses:
    - ip_netmask:
        get_param: TenantIpSubnet
    routes:
    -
      ip_netmask:
        get_param: TenantSupernet
      next_hop:
        get_param: TenantInterfaceDefaultRoute

ComputeKernel0Hw1_network_config:
  - type: interface
    name: nic1
    dns_servers:
      get_param: DnsServers
    use_dhcp: false
    mtu:
      get_param: ControlPlaneMtu
    addresses:
    - ip_netmask:
        list_join:
          - '/'
          - - get_param: ControlPlaneIp
            - get_param: ControlPlaneSubnetCidr
```

```
    routes:
    -
      default: True
      next_hop:
        get_param: ControlPlaneDefaultRoute
  - type: vlan
    device: nic1
    vlan_id:
      get_param: Storage0NetworkVlanID
    mtu:
      get_param: Storage0NetworkMtu
    addresses:
    - ip_netmask:
        get_param: Storage0IpSubnet
    routes:
    -
      ip_netmask:
        get_param: StorageSupernet
      next_hop:
        get_param: Storage0InterfaceDefaultRoute
  - type: vlan
    device: nic1
    vlan_id:
      get_param: InternalApi0NetworkVlanID
    mtu:
      get_param: InternalApi0NetworkMtu
    addresses:
    - ip_netmask:
        get_param: InternalApi0IpSubnet
    routes:
    -
      ip_netmask:
        get_param: InternalApiSupernet
      next_hop:
        get_param: InternalApi0InterfaceDefaultRoute
  - type: interface
    name: nic2
    use_dhcp: false
  - type: linux_bond
    name: bond0
    use_dhcp: false
    bonding_options: "mode=802.3ad xmit_hash_policy=layer3+4 lacp_rate=fast updelay=1000
miimon=100"
```

```
    mtu:
      get_param: Tenant0NetworkMtu
    members:
    - type: interface
      name: nic3
      primary: true
      mtu:
        get_param: Tenant0NetworkMtu
    - type: interface
      name: nic4
      mtu:
        get_param: Tenant0NetworkMtu
  - type: vlan
    vlan_id:
      get_param: Tenant0NetworkVlanID
    device: bond0
  - type: contrail_vrouter
    name: vhost0
    use_dhcp: false
    members:
      -
        type: interface
        name:
          str_replace:
            template: vlanVLANID
            params:
              VLANID: {get_param: Tenant0NetworkVlanID}
        use_dhcp: false
    addresses:
    - ip_netmask:
        get_param: Tenant0IpSubnet
    mtu:
      get_param: Tenant0NetworkMtu
    routes:
    -
      ip_netmask:
        get_param: TenantSupernet
      next_hop:
        get_param: Tenant0InterfaceDefaultRoute

ComputeKernel0Hw0_network_config:
  - type: interface
    name: nic1
```

```
    dns_servers:
      get_param: DnsServers
    use_dhcp: false
    mtu:
      get_param: ControlPlaneMtu
    addresses:
    - ip_netmask:
        list_join:
          - '/'
          - - get_param: ControlPlaneIp
            - get_param: ControlPlaneSubnetCidr
    routes:
    -
      default: True
      next_hop:
        get_param: ControlPlaneDefaultRoute
  - type: vlan
    device: nic1
    vlan_id:
      get_param: Storage0NetworkVlanID
    mtu:
      get_param: Storage0NetworkMtu
    addresses:
    - ip_netmask:
        get_param: Storage0IpSubnet
    routes:
    -
      ip_netmask:
        get_param: StorageSupernet
      next_hop:
        get_param: Storage0InterfaceDefaultRoute
  - type: vlan
    device: nic1
    vlan_id:
      get_param: InternalApi0NetworkVlanID
    mtu:
      get_param: InternalApi0NetworkMtu
    addresses:
    - ip_netmask:
        get_param: InternalApi0IpSubnet
    routes:
    -
      ip_netmask:
```

```
        get_param: InternalApiSupernet
      next_hop:
        get_param: InternalApi0InterfaceDefaultRoute
  - type: interface
    name: nic2
    use_dhcp: false
  - type: linux_bond
    name: bond0
    use_dhcp: false
    bonding_options: "mode=802.3ad xmit_hash_policy=layer3+4 lacp_rate=fast updelay=1000
miimon=100"
    mtu:
      get_param: Tenant0NetworkMtu
    members:
    - type: interface
      name: nic3
      primary: true
      mtu:
        get_param: Tenant0NetworkMtu
    - type: interface
      name: nic4
      mtu:
        get_param: Tenant0NetworkMtu
  - type: vlan
    vlan_id:
      get_param: Tenant0NetworkVlanID
    device: bond0
  - type: contrail_vrouter
    name: vhost0
    use_dhcp: false
    members:
      -
        type: interface
        name:
          str_replace:
            template: vlanVLANID
            params:
              VLANID: {get_param: Tenant0NetworkVlanID}
        use_dhcp: false
    addresses:
    - ip_netmask:
        get_param: Tenant0IpSubnet
    mtu:
```

```
      get_param: Tenant0NetworkMtu
    routes:
    -
      ip_netmask:
        get_param: TenantSupernet
      next_hop:
        get_param: Tenant0InterfaceDefaultRoute

ComputeKernel1Hw0_network_config:
  - type: interface
    name: nic1
    dns_servers:
      get_param: DnsServers
    use_dhcp: false
    mtu:
      get_param: ControlPlaneMtu
    addresses:
    - ip_netmask:
        list_join:
          - '/'
          - - get_param: ControlPlaneIp
            - get_param: ControlPlaneSubnetCidr
    routes:
    -
      default: True
      next_hop:
        get_param: ControlPlaneDefaultRoute
  - type: vlan
    device: nic1
    vlan_id:
      get_param: Storage1NetworkVlanID
    mtu:
      get_param: Storage1NetworkMtu
    addresses:
    - ip_netmask:
        get_param: Storage1IpSubnet
    routes:
    -
      ip_netmask:
        get_param: StorageSupernet
      next_hop:
        get_param: Storage1InterfaceDefaultRoute
  - type: vlan
```

```
    device: nic1
    vlan_id:
      get_param: InternalApi1NetworkVlanID
    mtu:
      get_param: InternalApi1NetworkMtu
    addresses:
    - ip_netmask:
        get_param: InternalApi1IpSubnet
    routes:
    -
      ip_netmask:
        get_param: InternalApiSupernet
      next_hop:
        get_param: InternalApi1InterfaceDefaultRoute
  - type: interface
    name: nic2
    use_dhcp: false
  - type: linux_bond
    name: bond0
    use_dhcp: false
    bonding_options: "mode=802.3ad xmit_hash_policy=layer3+4 lacp_rate=fast updelay=1000
miimon=100"
    mtu:
      get_param: Tenant1NetworkMtu
    members:
    - type: interface
      name: nic3
      primary: true
      mtu:
        get_param: Tenant1NetworkMtu
    - type: interface
      name: nic4
      mtu:
        get_param: Tenant1NetworkMtu
  - type: vlan
    vlan_id:
      get_param: Tenant1NetworkVlanID
    device: bond0
  - type: contrail_vrouter
    name: vhost0
    use_dhcp: false
    members:
      -
```

```
        type: interface
        name:
          str_replace:
            template: vlanVLANID
            params:
              VLANID: {get_param: Tenant1NetworkVlanID}
        use_dhcp: false
    addresses:
    - ip_netmask:
        get_param: Tenant1IpSubnet
    mtu:
      get_param: Tenant1NetworkMtu
    routes:
    -
      ip_netmask:
        get_param: TenantSupernet
      next_hop:
        get_param: Tenant1InterfaceDefaultRoute

ComputeKernel1Hw1_network_config:
  - type: interface
    name: nic1
    dns_servers:
      get_param: DnsServers
    use_dhcp: false
    mtu:
      get_param: ControlPlaneMtu
    addresses:
    - ip_netmask:
        list_join:
          - '/'
          - - get_param: ControlPlaneIp
            - get_param: ControlPlaneSubnetCidr
    routes:
    -
      default: True
      next_hop:
        get_param: ControlPlaneDefaultRoute
  - type: vlan
    device: nic1
    vlan_id:
      get_param: Storage1NetworkVlanID
    mtu:
```

```
      get_param: Storage1NetworkMtu
    addresses:
    - ip_netmask:
        get_param: Storage1IpSubnet
    routes:
    -
      ip_netmask:
        get_param: StorageSupernet
      next_hop:
        get_param: Storage1InterfaceDefaultRoute
  - type: vlan
    device: nic1
    vlan_id:
      get_param: InternalApi1NetworkVlanID
    mtu:
      get_param: InternalApi1NetworkMtu
    addresses:
    - ip_netmask:
        get_param: InternalApi1IpSubnet
    routes:
    -
      ip_netmask:
        get_param: InternalApiSupernet
      next_hop:
        get_param: InternalApi1InterfaceDefaultRoute
  - type: interface
    name: nic2
    use_dhcp: false
  - type: linux_bond
    name: bond0
    use_dhcp: false
    bonding_options: "mode=802.3ad xmit_hash_policy=layer3+4 lacp_rate=fast updelay=1000
miimon=100"
    mtu:
      get_param: Tenant1NetworkMtu
    members:
    - type: interface
      name: nic3
      primary: true
      mtu:
        get_param: Tenant1NetworkMtu
    - type: interface
      name: nic4
```

```
      mtu:
        get_param: Tenant1NetworkMtu
  - type: vlan
    vlan_id:
      get_param: Tenant1NetworkVlanID
    device: bond0
  - type: contrail_vrouter
    name: vhost0
    use_dhcp: false
    members:
      -
        type: interface
        name:
          str_replace:
            template: vlanVLANID
            params:
              VLANID: {get_param: Tenant1NetworkVlanID}
        use_dhcp: false
    addresses:
    - ip_netmask:
        get_param: Tenant1IpSubnet
    mtu:
      get_param: Tenant1NetworkMtu
    routes:
    -
      ip_netmask:
        get_param: TenantSupernet
      next_hop:
        get_param: Tenant1InterfaceDefaultRoute

ComputeSriov0Hw4_network_config:
  - type: interface
    name: nic1
    dns_servers:
      get_param: DnsServers
    use_dhcp: false
    mtu:
      get_param: ControlPlaneMtu
    addresses:
    - ip_netmask:
        list_join:
          - '/'
          - - get_param: ControlPlaneIp
```

```
            - get_param: ControlPlaneSubnetCidr
    routes:
    -
      default: True
      next_hop:
        get_param: ControlPlaneDefaultRoute
  - type: vlan
    device: nic1
    vlan_id:
      get_param: Storage0NetworkVlanID
    mtu:
      get_param: Storage0NetworkMtu
    addresses:
    - ip_netmask:
        get_param: Storage0IpSubnet
    routes:
    -
      ip_netmask:
        get_param: StorageSupernet
      next_hop:
        get_param: Storage0InterfaceDefaultRoute
  - type: vlan
    device: nic1
    vlan_id:
      get_param: InternalApi0NetworkVlanID
    mtu:
      get_param: InternalApi0NetworkMtu
    addresses:
    - ip_netmask:
        get_param: InternalApi0IpSubnet
    routes:
    -
      ip_netmask:
        get_param: InternalApiSupernet
      next_hop:
        get_param: InternalApi0InterfaceDefaultRoute
  - type: interface
    name: nic2
    use_dhcp: false
  - type: linux_bond
    name: bond0
    use_dhcp: false
    bonding_options: "mode=802.3ad xmit_hash_policy=layer3+4 lacp_rate=fast updelay=1000
```

```
miimon=100"
    mtu:
      get_param: Tenant0NetworkMtu
    members:
    - type: interface
      name: nic3
      primary: true
      mtu:
        get_param: Tenant0NetworkMtu
    - type: interface
      name: nic4
      mtu:
        get_param: Tenant0NetworkMtu
  - type: vlan
    vlan_id:
      get_param: Tenant0NetworkVlanID
    device: bond0
  - type: contrail_vrouter
    name: vhost0
    use_dhcp: false
    members:
      -
        type: interface
        name:
          str_replace:
            template: vlanVLANID
            params:
              VLANID: {get_param: Tenant0NetworkVlanID}
        use_dhcp: false
    addresses:
    - ip_netmask:
        get_param: Tenant0IpSubnet
    mtu:
      get_param: Tenant0NetworkMtu
    routes:
    -
      ip_netmask:
        get_param: TenantSupernet
      next_hop:
        get_param: Tenant0InterfaceDefaultRoute

ComputeSriov1Hw5_network_config:
  - type: interface
```

```
    name: nic1
    dns_servers:
      get_param: DnsServers
    use_dhcp: false
    mtu:
      get_param: ControlPlaneMtu
    addresses:
    - ip_netmask:
        list_join:
          - '/'
          - - get_param: ControlPlaneIp
            - get_param: ControlPlaneSubnetCidr
    routes:
    -
      default: True
      next_hop:
        get_param: ControlPlaneDefaultRoute
  - type: vlan
    device: nic1
    vlan_id:
      get_param: Storage1NetworkVlanID
    mtu:
      get_param: Storage1NetworkMtu
    addresses:
    - ip_netmask:
        get_param: Storage1IpSubnet
    routes:
    -
      ip_netmask:
        get_param: StorageSupernet
      next_hop:
        get_param: Storage1InterfaceDefaultRoute
  - type: vlan
    device: nic1
    vlan_id:
      get_param: InternalApi1NetworkVlanID
    mtu:
      get_param: InternalApi1NetworkMtu
    addresses:
    - ip_netmask:
        get_param: InternalApi1IpSubnet
    routes:
    -
```

```
      ip_netmask:
        get_param: InternalApiSupernet
      next_hop:
        get_param: InternalApi1InterfaceDefaultRoute
  - type: interface
    name: nic2
    use_dhcp: false
  - type: linux_bond
    name: bond0
    use_dhcp: false
    bonding_options: "mode=802.3ad xmit_hash_policy=layer3+4 lacp_rate=fast updelay=1000
miimon=100"
    mtu:
      get_param: Tenant1NetworkMtu
    members:
    - type: interface
      name: nic3
      primary: true
      mtu:
        get_param: Tenant1NetworkMtu
    - type: interface
      name: nic4
      mtu:
        get_param: Tenant1NetworkMtu
  - type: vlan
    vlan_id:
      get_param: Tenant1NetworkVlanID
    device: bond0
  - type: contrail_vrouter
    name: vhost0
    use_dhcp: false
    members:
      -
        type: interface
        name:
          str_replace:
            template: vlanVLANID
            params:
              VLANID: {get_param: Tenant1NetworkVlanID}
        use_dhcp: false
    addresses:
    - ip_netmask:
        get_param: Tenant1IpSubnet
```

```
    mtu:
      get_param: Tenant1NetworkMtu
    routes:
    -
      ip_netmask:
        get_param: TenantSupernet
      next_hop:
        get_param: Tenant1InterfaceDefaultRoute

ComputeDpdk0Hw2_network_config:
  - type: interface
    name: nic1
    dns_servers:
      get_param: DnsServers
    use_dhcp: false
    mtu:
      get_param: ControlPlaneMtu
    addresses:
    - ip_netmask:
        list_join:
          - '/'
          - - get_param: ControlPlaneIp
            - get_param: ControlPlaneSubnetCidr
    routes:
    -
      default: True
      next_hop:
        get_param: ControlPlaneDefaultRoute
  - type: vlan
    device: nic1
    vlan_id:
      get_param: Storage0NetworkVlanID
    mtu:
      get_param: Storage0NetworkMtu
    addresses:
    - ip_netmask:
        get_param: Storage0IpSubnet
    routes:
    -
      ip_netmask:
        get_param: StorageSupernet
      next_hop:
        get_param: Storage0InterfaceDefaultRoute
```

```
  - type: vlan
    device: nic1
    vlan_id:
      get_param: InternalApi0NetworkVlanID
    mtu:
      get_param: InternalApi0NetworkMtu
    addresses:
    - ip_netmask:
        get_param: InternalApi0IpSubnet
    routes:
    -
      ip_netmask:
        get_param: InternalApiSupernet
      next_hop:
        get_param: InternalApi0InterfaceDefaultRoute
  - type: interface
    name: nic2
    use_dhcp: false
  - type: contrail_vrouter_dpdk
    name: vhost0
    vlan_id:
      get_param: Tenant0NetworkVlanID
    driver: "{{ overcloud['contrail']['vrouter']['dpdk']['driver'] }}"
    bond_mode: 4
    bond_policy: layer2+3
    cpu_list: 1,2
    members:
    - type: interface
      name: nic3
    - type: interface
      name: nic4
    addresses:
    - ip_netmask:
        get_param: Tenant0IpSubnet
    mtu:
      get_param: Tenant0NetworkMtu
    routes:
    -
      ip_netmask:
        get_param: TenantSupernet
      next_hop:
        get_param: Tenant0InterfaceDefaultRoute
```

```
ComputeDpdk1Hw3_network_config:
  - type: interface
    name: nic1
    dns_servers:
      get_param: DnsServers
    use_dhcp: false
    mtu:
      get_param: ControlPlaneMtu
    addresses:
    - ip_netmask:
        list_join:
          - '/'
          - - get_param: ControlPlaneIp
            - get_param: ControlPlaneSubnetCidr
    routes:
    -
      default: True
      next_hop:
        get_param: ControlPlaneDefaultRoute
  - type: vlan
    device: nic1
    vlan_id:
      get_param: Storage1NetworkVlanID
    mtu:
      get_param: Storage1NetworkMtu
    addresses:
    - ip_netmask:
        get_param: Storage1IpSubnet
    routes:
    -
      ip_netmask:
        get_param: StorageSupernet
      next_hop:
        get_param: Storage1InterfaceDefaultRoute
  - type: vlan
    device: nic1
    vlan_id:
      get_param: InternalApi1NetworkVlanID
    mtu:
      get_param: InternalApi1NetworkMtu
    addresses:
    - ip_netmask:
        get_param: InternalApi1IpSubnet
```

```
    routes:
    -
      ip_netmask:
        get_param: InternalApiSupernet
      next_hop:
        get_param: InternalApi1InterfaceDefaultRoute
  - type: interface
    name: nic2
    use_dhcp: false
  - type: contrail_vrouter_dpdk
    name: vhost0
    vlan_id:
      get_param: Tenant1NetworkVlanID
    driver: "{{ overcloud['contrail']['vrouter']['dpdk']['driver'] }}"
    bond_mode: 4
    bond_policy: layer2+3
    cpu_list: 1,2
    members:
    - type: interface
      name: nic3
    - type: interface
      name: nic4
    addresses:
    - ip_netmask:
        get_param: Tenant1IpSubnet
    mtu:
      get_param: Tenant1NetworkMtu
    routes:
    -
      ip_netmask:
        get_param: TenantSupernet
      next_hop:
        get_param: Tenant1InterfaceDefaultRoute

CephStorage0Hw6_network_config:
  - type: interface
    name: nic1
    dns_servers:
      get_param: DnsServers
    use_dhcp: false
    mtu:
      get_param: ControlPlaneMtu
    addresses:
```

```
    - ip_netmask:
        list_join:
          - '/'
          - - get_param: ControlPlaneIp
            - get_param: ControlPlaneSubnetCidr
      routes:
      -
        default: True
        next_hop:
          get_param: ControlPlaneDefaultRoute
    - type: vlan
      device: nic1
      vlan_id:
        get_param: Storage0NetworkVlanID
      mtu:
        get_param: Storage0NetworkMtu
      addresses:
      - ip_netmask:
          get_param: Storage0IpSubnet
      routes:
      -
        ip_netmask:
          get_param: StorageSupernet
        next_hop:
          get_param: Storage0InterfaceDefaultRoute
    - type: vlan
      device: nic1
      vlan_id:
        get_param: StorageMgmt0NetworkVlanID
      mtu:
        get_param: StorageMgmt0NetworkMtu
      addresses:
      - ip_netmask:
          get_param: StorageMgmt0IpSubnet
      routes:
      -
        ip_netmask:
          get_param: StorageMgmtSupernet
        next_hop:
          get_param: StorageMgmt0InterfaceDefaultRoute
    - type: interface
      name: nic2
      use_dhcp: false
```

```
  - type: interface
    name: nic3
    use_dhcp: false
  - type: interface
    name: nic4
    use_dhcp: false

CephStorage1Hw7_network_config:
  - type: interface
    name: nic1
    dns_servers:
      get_param: DnsServers
    use_dhcp: false
    mtu:
      get_param: ControlPlaneMtu
    addresses:
    - ip_netmask:
        list_join:
          - '/'
          - - get_param: ControlPlaneIp
            - get_param: ControlPlaneSubnetCidr
    routes:
    -
      default: True
      next_hop:
        get_param: ControlPlaneDefaultRoute
  - type: vlan
    device: nic1
    vlan_id:
      get_param: Storage1NetworkVlanID
    mtu:
      get_param: Storage1NetworkMtu
    addresses:
    - ip_netmask:
        get_param: Storage1IpSubnet
    routes:
    -
      ip_netmask:
        get_param: StorageSupernet
      next_hop:
        get_param: Storage1InterfaceDefaultRoute
  - type: vlan
    device: nic1
```

```
    vlan_id:
      get_param: StorageMgmt1NetworkVlanID
    mtu:
      get_param: StorageMgmt1NetworkMtu
    addresses:
    - ip_netmask:
        get_param: StorageMgmt1IpSubnet
    routes:
    -
      ip_netmask:
        get_param: StorageMgmtSupernet
      next_hop:
        get_param: StorageMgmt1InterfaceDefaultRoute
  - type: interface
    name: nic2
    use_dhcp: false
  - type: interface
    name: nic3
    use_dhcp: false
  - type: interface
    name: nic4
    use_dhcp: false
```

Sample compute-nodes.yml Configuration File

```
# Copyright 2018 Juniper Networks, Inc. All rights reserved.
# Licensed under the Juniper Networks Script Software License (the "License").
# You may not use this script file except in compliance with the License, which is located at
# http://www.juniper.net/support/legal/scriptlicense/
# Unless required by applicable law or otherwise agreed to in writing by the parties,
# software distributed under the License is distributed on an "AS IS" BASIS,
# WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
#
#
#

# Each list item contains:
#    name: name of a node in the inventory (string)
#    profile: name of hardware profile, group of servers (optional, string)
```

```
#    leaf: leaf name (optional, string)

# List of nodes to use as compute role using Contrail DPDK vRouter
compute_nodes_dpdk:
  - name: computedpdk1
    leaf: '0'
    profile: hw2
  - name: computedpdk2
    leaf: '1'
    profile: hw3

# List of nodes to use as compute role using Sriov
compute_nodes_sriov:
  - name: computesriov1
    leaf: '0'
    profile: hw4
  - name: computesriov2
    leaf: '1'
    profile: hw5

# List of nodes to use as compute role using Contrail kernel vRouter
compute_nodes_kernel:
  - name: compute1
    leaf: '0'
    profile: hw0
  - name: compute2
    leaf: '0'
    profile: hw1
  - name: compute3
    leaf: '1'
    profile: hw1
  - name: compute4
    leaf: '1'
    profile: hw0
```

Sample storage-nodes.yml Configuration File

```
# Copyright 2018 Juniper Networks, Inc. All rights reserved.
# Licensed under the Juniper Networks Script Software License (the "License").
```

```
# You may not use this script file except in compliance with the License, which is located at
# http://www.juniper.net/support/legal/scriptlicense/
# Unless required by applicable law or otherwise agreed to in writing by the parties,
# software distributed under the License is distributed on an "AS IS" BASIS,
# WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
#
#
#

# List of nodes to use as storage host role
# List item contains:
#    name: name of a node in the inventory (string)
#    profile: name of hardware profile, group of servers (optional, string)
#    leaf: leaf name (optional, string)

storage_nodes:
  - name: storage1
    leaf: '0'
    profile: hw6
  - name: storage2
    leaf: '0'
    profile: hw6
  - name: storage3
    leaf: '1'
    profile: hw7
```

Sample k8s-host-nodes.yml Configuration File

```
# Copyright 2020 Juniper Networks, Inc. All rights reserved.
# Licensed under the Juniper Networks Script Software License (the "License").
# You may not use this script file except in compliance with the License, which is located at
# http://www.juniper.net/support/legal/scriptlicense/
# Unless required by applicable law or otherwise agreed to in writing by the parties,
# software distributed under the License is distributed on an "AS IS" BASIS,
# WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
#
#
# +-------------------+   +--------------+    +-----------------+
# |Undercloud/Jumphost|   |OpenStack Ctrl|    |OpenStack Compute|
```

```
# +-------------------+   +--------------+    +-----------------+
# |Provision & deploy |   |Neutron       |    |vrouter          |
# +-+-----------------+   |Heat          |    |                 |
#   |                     |Keystone      |    |                 |
#   |                     ++-------------+    |                 |
#   |                      |                  ++--+-------------+
#   |                      |                   |  |
#   |                      |                   |  |
#   |                      |                   |  +----+-------------------------------------
tenant_ip_netmask------------+
#   |                      |                   |       |
#   |                      |                   |       |
#   |                       +------------+------+---------------------+-------------+---------
internalapi_ip_netmask-------+
#   |                                    |             |             |            |
#   |                                    |             |             |            |
#   |                        +-----------------------------------------------------------+
+-----------+   +-----------+
#   |                         |          |             |             |            |      |
|           |   |           |
#   +--control_ip_netmask----+ +--------+--+   +-------+--+   +-------+--+  +-------+--+   |
|           |   |           |
#                             | |Config     |   |Control   |   |Command   |  |K8s api   |   |
|           |   |           |
#                             | +-----------+   +----------+   +----------+  +-+--------+   |
|           |   |           |
#                             | |Analytics  |                 |WebUI     |    |            |
|           |   |           |
#                             | +-----------+                 +-----+----+    |            |
|           |   |           |
#                             |                                     |        |            |
|           |   |           |
#                             |                                  +--+---------+--+         |
|           |   |           |
#                             |                K8s host          |  VIP & PROXY  |         |
|  K8s host |   |  K8s host |
#                             +----------------------------------+---------------+---------+
+-----------+   +-----------+
#                                                                |         |
#                                                                |         |
#                                                              +---+---------+---------------
external_ip_netmask----------+
#
```

```
# Contrail control plane deployed on Kubernetes.
#
# Every node defined in k8s_host_nodes list should have following keys defined:
#  * name: hostname without dns domain. Later it will be concatenated with
#          global['domain'] value from site.yml.
#
#  * hypervisor: Name of control host defined in control-host-nodes.yml where the
#                k8s nodes VM will be hosted.
#
#  * control_ip_netmask: IP address in CIDR notation in control network.
#                        This network will be used to provision, deploy and access
#                        node.
#
#  * internalapi_ip_netmask: IP address in CIDR notation in internal api network.
#                            This network will be used for communication between
#                            OpenStack components like Neutron and Heat and
#                            Contrail services. Contrail vrouter will use this
#                            network to provision itself. K8s components utilises
#                            this network for internal communication.
#
#  * tenant_ip_netmask: IP address in CIDR notation in tenant network.
#                       Through this network vrouter will communicate with
#                       Contrail control.
#
#  * external_ip_netmask: IP address in CIDR notation in external network.
#                          This network will be used to setup external VIP
#                          managed by keepalived. Haproxy will be configured
#                          to expose services like Command, Contrail web ui and
#                          k8s api through this VIP.
#
#  * extra keys: nodes can define variables used later in network configuration
#                like dns servers, default gw etc.
common: &common
  dns_server1: "1.1.1.1"
  dns_server2: "8.8.8.8"
  gw: "192.2.0.254"

k8s_host_nodes:
  - name: k8s-contrail1
    hypervisor: controler1
    control_ip_netmask: "192.168.213.21/24"
    internalapi_ip_netmask: "172.16.0.21/24"
    tenant_ip_netmask: "172.17.131.21/24"
```

```
    external_ip_netmask: "192.2.0.230/25"
    <<: *common
  - name: k8s-contrail2
    hypervisor: controler2
    control_ip_netmask: "192.168.213.22/24"
    internalapi_ip_netmask: "172.16.0.22/24"
    tenant_ip_netmask: "172.17.131.22/24"
    external_ip_netmask: "192.2.0.231/25"
    <<: *common
  - name: k8s-contrail3
    hypervisor: controler3
    control_ip_netmask: "192.168.213.23/24"
    internalapi_ip_netmask: "172.16.0.23/24"
    tenant_ip_netmask: "172.17.131.23/24"
    external_ip_netmask: "192.2.0.232/25"
    <<: *common

# Template for network layout on all control host nodes
# This follows the os-net-config syntax
# See https://github.com/openstack/os-net-config/tree/stable/queens
k8s_host_nodes_network_config:
  - type: interface
    name: nic1
    addresses:
      - ip_netmask: "{{ host.control_ip_netmask }}"
  - type: vlan
    device: nic1
    vlan_id: "{{ overcloud.network.internal_api.vlan }}"
    addresses:
      - ip_netmask: "{{ host.internalapi_ip_netmask }}"
    routes:
      - ip_netmask: "{{ overcloud.network.internal_api.supernet }}"
        next_hop: "{{ overcloud.network.internal_api.gateway }}"
  - type: interface
    name: nic2
    # Ensure that resolv.conf keeps nameservers
    nm_controlled: true
    addresses:
      - ip_netmask: "{{ host.external_ip_netmask }}"
    routes:
      - next_hop: "{{ host.gw }}"
        default: true
    dns_servers:
```

```
      - "{{ host.dns_server1 }}"
      - "{{ host.dns_server2 }}"
  - type: interface
    name: nic3
    use_dhcp: false
  - type: vlan
    device: nic3
    # For now discover of tenant network IP by contrail control
    # does not work when interface is controlled by NetworkManager
    nm_controlled: false
    vlan_id: "{{ overcloud.network.tenant.vlan }}"
    addresses:
      - ip_netmask: "{{ host.tenant_ip_netmask }}"
    routes:
      - ip_netmask: "{{ overcloud.network.tenant.supernet }}"
        next_hop: "{{ overcloud.network.tenant.gateway }}"
```

Sample vault-data.yml Configuration File

```
# This config structure can be used to hold information that needs to be encrypted for privacy
# If there is a password stored in /var/lib/contrail_cloud/config/.vault_password then it will
be used
# Otherwise the password can be entered interactively
#
# This file can be edited with the "ansible-vault edit" command
# This file can be re-encrypted with a new password with the "ansible-vault rekey" command
vault:
  global:
    rhel:
      # Contrail Cloud Activation Key
      satellite:
        #SATELLITE_KEY
        key: "PUT_YOUR_KEY_HERE"
    # User account used for all Contrail Cloud automation
    # This account will be created on:
    #    - jumphost
    #    - control hosts
    #    - all overcloud roles
    #    - appformix controllers
```

```
  service_user:
    # Account Name
    name: "contrail"
    # Account Password
    password: "c0ntrail123"
    # Passphrase used to encrypt ssh key of service user.
    # If not defined ssh private key will not be encrypted.
    # ssh_key_passphrase: "c0ntrail123"
rhvm:
  vm:
    # rhvm user name
    user: "contrail"
    # password for the rhvm vm user
    password: "c0ntrail123"
    # root password for the rhvm VM
    root_password: "c0ntrail123"
    # keystone admin password
    admin_password: "c0ntrail123"
    # Passphrase used to encrypt ssh key of rhvm user.
    # If not defined ssh private key will not be encrypted.
    # ssh_key_passphrase: "c0ntrail123"
    vnc:
      # VNC console password for the rhvm VM
      password: "contrail123"
undercloud:
  #Administrator password - default is randomly generated
  #admin_password: "c0ntrail123"
  vm:
    # undercloud user name
    user: "stack"
    # password for the undercloud vm user
    password: "contrail123"
    # root password for the undercloud VM
    root_password: "contrail123"
    # Passphrase used to encrypt ssh key of undercloud user.
    # If not defined ssh private key will not be encrypted.
    # ssh_key_passphrase: "c0ntrail123"
    vnc:
      # VNC console password for the undercloud VM
      password: "contrail123"
overcloud:
  #Administrator password
  admin_password: "c0ntrail123"
```

```
  # Root password used for local login to overcloud nodes through console
  # root_password: "contrail123"
  contrail:
    rabbitmq:
      # contrail rabbitmq user name
      user: "contrail_rabbitmq"
      # contrail rabbitmq user password
      password: "c0ntrail123"
control_hosts:
  vm:
    vnc:
      # VNC console password for all control VMs
      password: "contrail123"
appformix:
  mysql:
    # Approfmix MySQL user account
    user: "appformix"
    # Approfmix MySQL user password
    password: "c0ntrail123"
  rabbitmq:
    # Approfmix RabbitMQ user account
    user: "appformix"
    # Approfmix RabbitMQ user password
    password: "c0ntrail123"
# Credentials used to connect external ceph cluster
#ceph_external:
#  client_key: "CLIENT_KEY"
#  client_user: "openstack"

# List of inventory hardware types that can hold hardware-specific properties
# You can create similar configutations to allow reference from inventory-nodes.yml
inventory_nodes:
  # A sample configuration for a hardware type
  hardware1:
    # IPMI user account for Ironic inevntory resources
    pm_user: "ADMIN"
    # IPMI user password for Ironic inevntory resources
    pm_password: "ADMIN"
  # A sample configuration for a hardware type
  hardware2:
    # IPMI user account for Ironic inevntory resource
    pm_user: "admin"
    # IPMI user password for Ironic inevntory resource
```

```
    pm_password: "admin"
# User defined sensitive data can be stored under 'other' key.
# Schema validation will only check if key,value format is used.
#other:
#  mykey: myvalue
```

RELATED DOCUMENTATION

Configuration File Changes | 25

No Link Title