Contrail plugin for Fuel Documentation

Release 5.1-5.1.0-1

Mirantis Inc.

Mar 21, 2017

Contents

1 Introduction **1**
- 1.1 Key terms, acronyms and abbreviations . . . 1
- 1.2 Overview . . . 1

2 New features in plugin version 5.1-5.1.0-1 **3**

3 Limitations **4**

4 Requirements **5**

5 Installation Guide **6**
- 5.1 Prerequisites . . . 6
- 5.2 Install Contrail Plugin . . . 6
- 5.3 Configure Contrail Plugin . . . 8

6 DPDK-based vRouter **15**
- 6.1 Description . . . 15
- 6.2 Prerequisites . . . 16
- 6.3 Restrictions . . . 16
- 6.4 Configure DPDK . . . 18
- 6.5 Verify DPDK . . . 22
- 6.6 Change DPDK options . . . 23
- 6.7 Change Huge Pages settings after deployment . . . 24

7 DPDK-based vRouter on virtual function (VF) **25**
- 7.1 Description . . . 25
- 7.2 Prerequisites . . . 25
- 7.3 How to enable DPDK on VF . . . 25

8 Enable SR-IOV **27**
- 8.1 Prerequisites . . . 27
- 8.2 Features . . . 27
- 8.3 SR-IOV Description . . . 27
- 8.4 Verify SR-IOV environment . . . 28
- 8.5 Enable SR-IOV in Fuel . . . 28
- 8.6 Create a virtual machine with SR-IOV device . . . 28

9 Contrail TSN **29**
9.1 TSN Description . . . 29
9.2 Prerequisites . . . 29
9.3 HA implementation details . . . 31
9.4 Configure TSN . . . 31

10 Contrail with VMware vCenter **33**
10.1 Overview . . . 33
10.2 Prerequisites . . . 34
10.3 Restrictions . . . 34
10.4 Configuration . . . 34
10.5 Verification . . . 37
10.6 VMware related options . . . 38
10.7 Add and delete ESXi hosts . . . 39

11 Contrail upgrades (experimental) **40**
11.1 Description . . . 40
11.2 Prequisites . . . 40
11.3 Update the packages on Fuel Master node . . . 40
11.4 Upgrade Contrail and OpenStack Controllers . . . 42
11.5 Upgrade Compute nodes . . . 42

12 Verify Contrail plugin **44**
12.1 Run OSTF tests . . . 46
12.2 Troubleshooting . . . 48

13 Restore failed Contrail node **50**

14 Use Contrail **52**
14.1 Log into Contrail . . . 53
14.2 Verify services status . . . 54
14.3 Create the virtual networks . . . 54

15 Appendix **56**

CHAPTER 1

Introduction

This document contains instructions for installing and configuring Contrail plugin for Fuel.

Key terms, acronyms and abbreviations

Juniper Contrail	Contrail Cloud Platform is a foundational element of Juniper's open cloud networking and NFV solutions.
SDN	Software defined network
RESTful API	Representational state transfer application programming interface
IDS	Intrusion detection system
DPI	Deep packet inspection
VIP	virtual IP address
BGP	Border gateway protocol
AS	Autonomous system
Contrail vRouter	Contrail vRouter is part of the compute node, which gets reachability information from the control plane and ensures native L3 services for host-based virtual machines.
MOS	Mirantis OpenStack
TOR	Top of rack
TSN	TOR Services Node

Overview

Contrail plugin for Fuel adds Contrail SDN to Mirantis OpenStack as a networking back end option using Fuel web UI in a user-friendly manner. Juniper Networks Contrail is an open software defined networking solution that automates and orchestrates the creation of highly scalable virtual networks.

Contrail features:

- Powerful API calls (REST or direct python class calls)
- Analytics engine: traffic flow reports, statistics
- Network management at 2-4 OSI layers
- Service chaining architecture: you can transparently pass traffic through service instances such as IDS, firewalls, and DPI.
- Fine grained virtual network access policy control

CHAPTER 2

New features in plugin version 5.1-5.1.0-1

- Fuel 9.2 with OpenStack Mitaka support
- Dedicated Analytics and Analytics DB node roles
- Contrail-Config, Contrail-Control and Contrail-DB are merged into Contrail-Controller role
- DPDK on VF, including bonded VFs
- Sahara and Murano compatibility
- vCenter as a secondary hypervisor support
- TSN in HA mode

CHAPTER 3

Limitations

- Plugin does not support removing Contrail-Controller and Analytics-DB nodes from a cluster. This can lead to data loss and must be a manual procedure. Plugin supports adding new Contrail-DB nodes to the environment.
- The Fuel Networking option "Assign public network to all nodes" is not compatible with Contrail Plugin.
- In case of using contrail service chaining with service instances, you may need to add *neutron* service user to a current project after you have deployed the environment:
  - Open OpenStack Dashboard, navigate to the *Identity* - *Projects* page.
  - Click *modify users* button on the right side of the `admin` project.
  - Add the `neutron` user to project members with `_member_` role.
- Changing the default OpenStack project name is not supported. Default project name should be `admin`.
- The password of OpenStack `admin` user should not contain following characters: `$`, `` ` ``, \\ and `!`
- Upgrade procedure based on custom deployment graphs does not support upgrading nodes with roles dpdk, tsn and vmware. Contrail packages upgrades should be done manually on these nodes.

CHAPTER 4

Requirements

The plugin has the following requirements for software and hardware:

Fuel version	9.2
Juniper Contrail version	3.2.0.0 testing was performed on 3.2.0.0-19
Hardware	• At least 2 additional servers: for Contrail controller and for combined Analytics and Analytics-DB Analytics-DB requires 320 Gb disk space. • Additional network interface on each node except for MOS Controller.

CHAPTER 5

Installation Guide

Prerequisites

This guide assumes that you have installed Fuel and all the nodes of your future environment are discovered and functional.

Install Contrail Plugin

To install the Contrail plugin:

1. Download the Contrail plugin from the Fuel Plugins Catalog.
2. Copy the rpm package downloaded at the previous step to the Fuel Master node and install the plugin

   ```
   scp contrail-5.1-5.1.0-1.noarch.rpm  <Fuel Master node ip>:/tmp/
   ```

3. Log into the Fuel Master node and install the plugin

   ```
   ssh <the Fuel Master node ip>
   fuel plugins --install contrail-5.1-5.1.0-1.noarch.rpm
   ```

   You should get the following output

   ```
   Plugin <plugin-name-version>.rpm was successfully installed
   ```

4. Copy the Juniper Contrail installation package to the Fuel Master node and run the installation script to unpack the vendor package and populate the plugin repository:

   Note: You can obtain the Juniper Contrail installation package from Juniper by subscription. More information can be found on the official Juniper Contrail web-site.

```
scp contrail-install-packages_3.2.0.0-19~mitaka_all.deb \
<Fuel Master node ip>:/var/www/nailgun/plugins/contrail-5.1/
ssh <Fuel Master node ip> /var/www/nailgun/plugins/contrail-5.1/install.sh
```

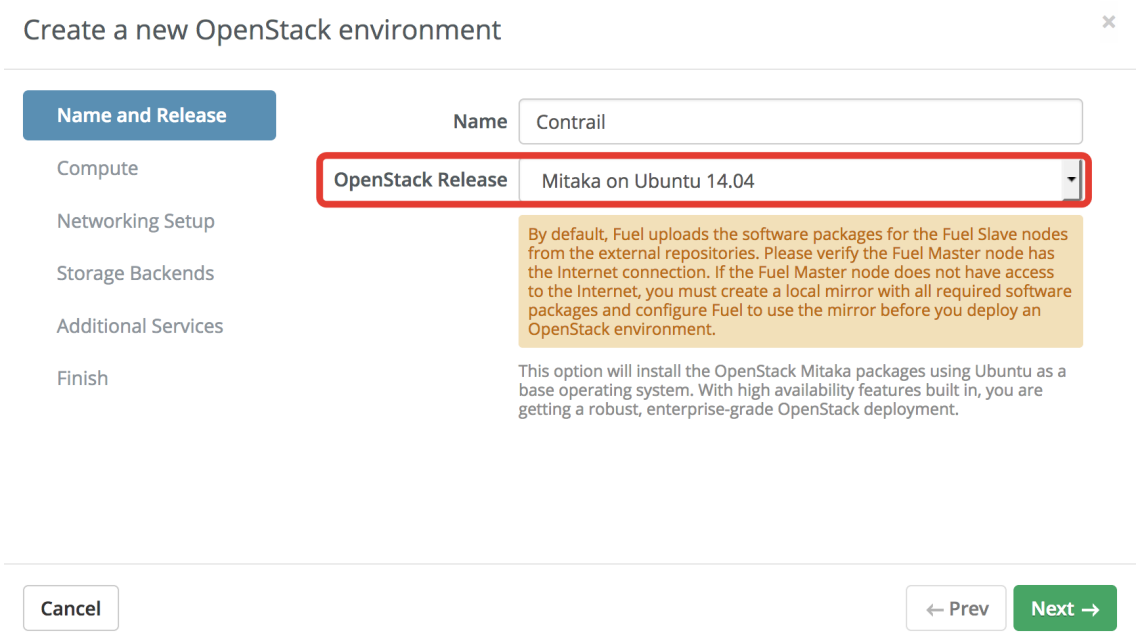
5. In case if vmware integration expected, before running *install.sh* copy vcenter plugin package to contrail plugin folder

```
scp contrail-install-vcenter-plugin_3.2.0.0-19_all.deb \
<Fuel Master node ip>:/var/www/nailgun/plugins/contrail-5.1/
```

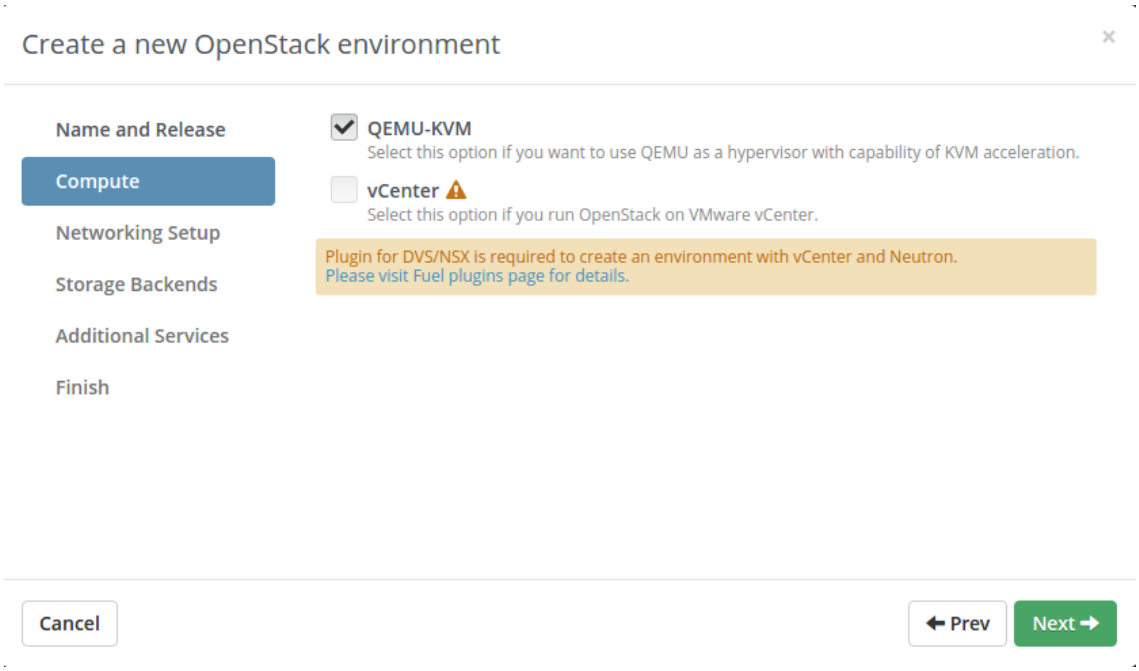
Configure Contrail Plugin

To configure the Contrail plugin, follow the steps below:

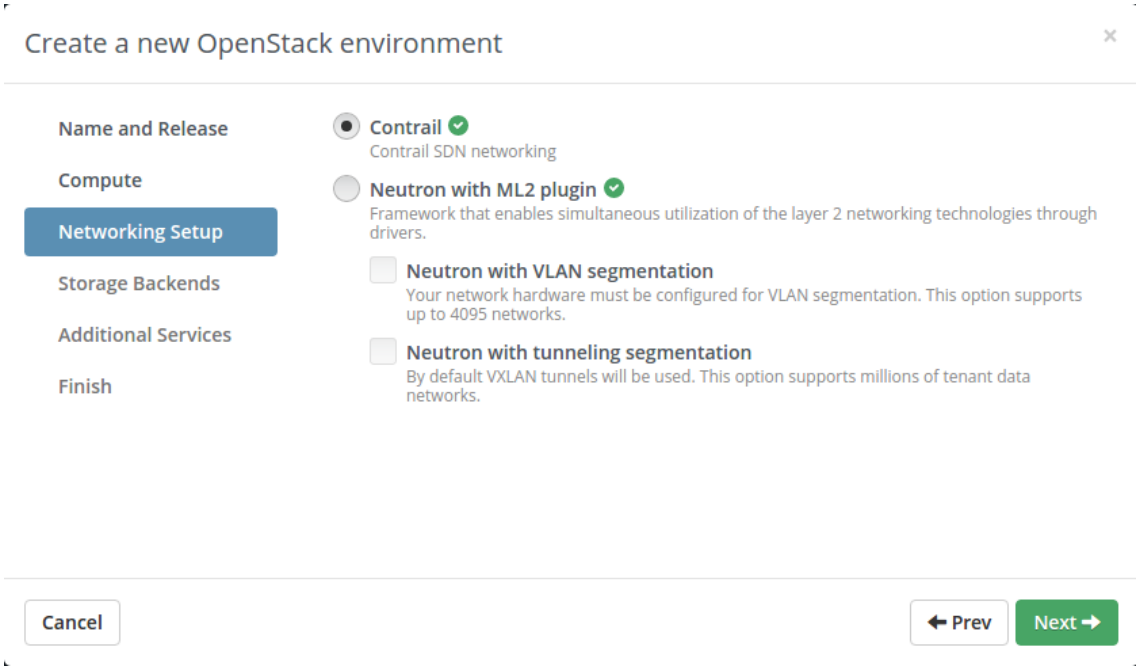
1. Create environment in Fuel web UI.



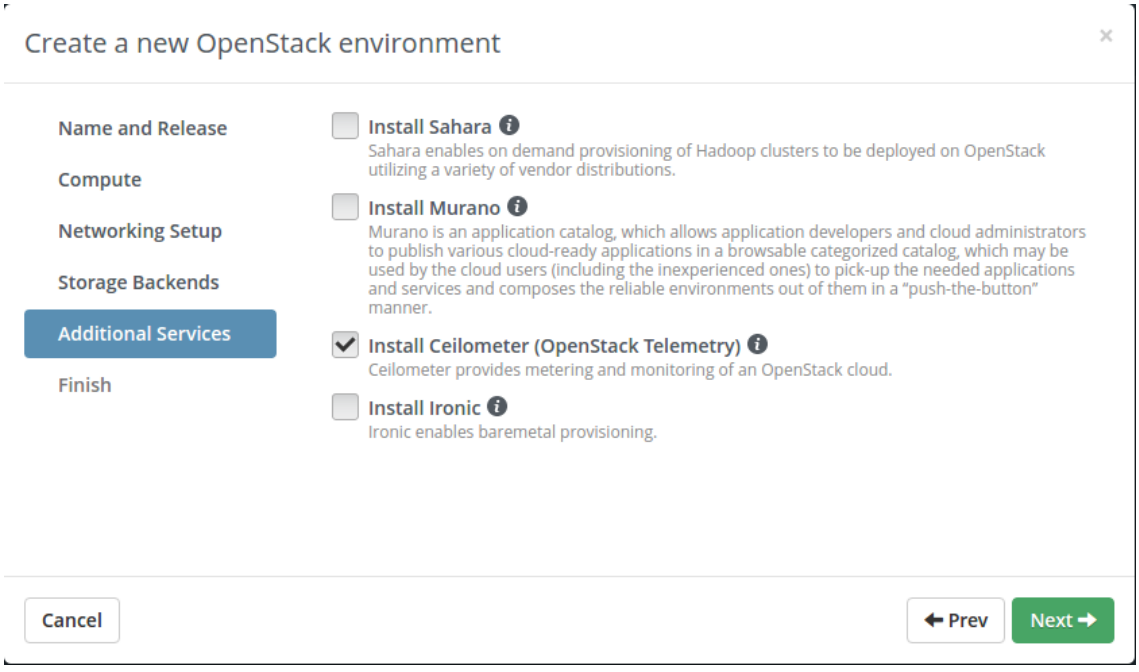
2. Please select KVM or QEMU hypervisor type for your environment also enable "vCenter" feature if vmware integration expected



3. Please select Contrail SDN networking setup.



4. If you plan to use Heat orchestration with autoscaling, install Ceilometer.



5. Enable the plugin and fill configuration fields with correct values:
   - AS number for BGP Gateway nodes communication defaults to 64512
   - IP addresses of gateway nodes provided as a comma-separated list - peer addresses for BGP interaction with border routers.

6. Add nodes and assign them the following roles:

   - At least 1 Controller
   - At least 1 Compute
   - At least 1 Contrail-Controller

     Note: Three or the greater odd number of nodes recommended for HA.

   - At least 1 Contrail-Analytics + Contrail-Analytics-DB

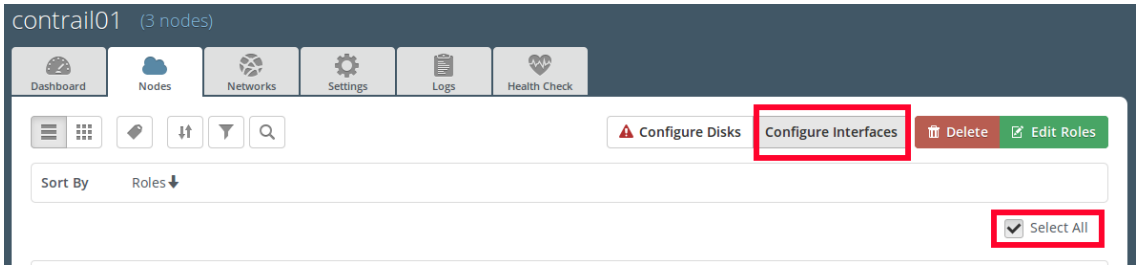
     Note: Three or the greater odd number of nodes recommended for HA.

   - If you plan to use Heat with autoscaling, in addition to Ceilometer you need to add node with MongoDB role

   The figure below shows sample node configuration.

7. The recommended size of partition for the Contrail-Analytics database is 256 GB or more.
8. Configure the network settings. See details at Fuel User Guide.

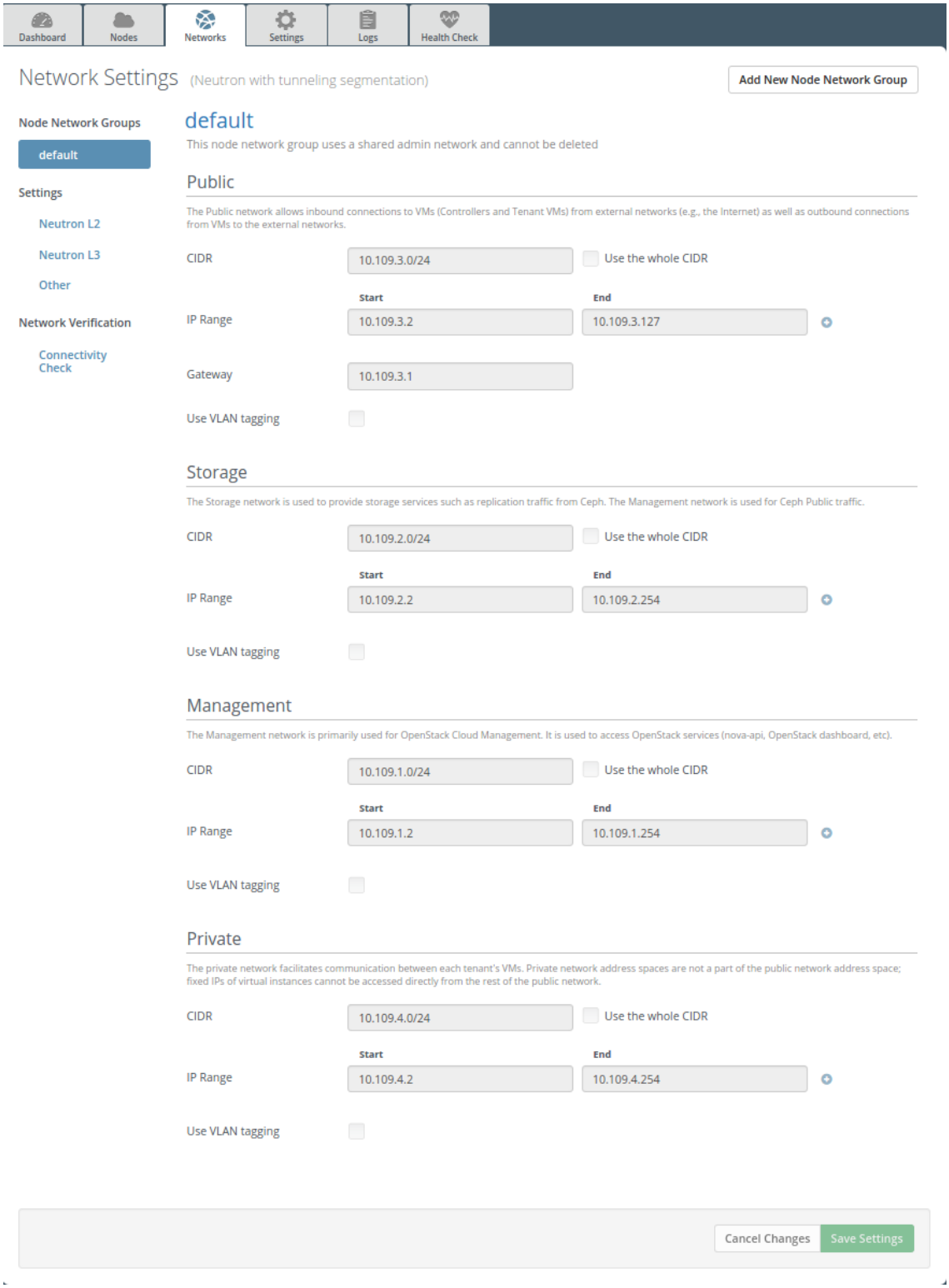
   Open *Nodes* tab, select all the nodes and press *Configure interfaces* button



   Set Private network to the separate network interface.

Warning: Do not use this physical interface for any other network.

Contrail vRouter will use this interface. Set the bigger MTU for Private interfaces, for example 9000, if switching hardware supports Jumbo Frames. This will enhance contrail network performance by avoiding packet fragmentation within Private network.

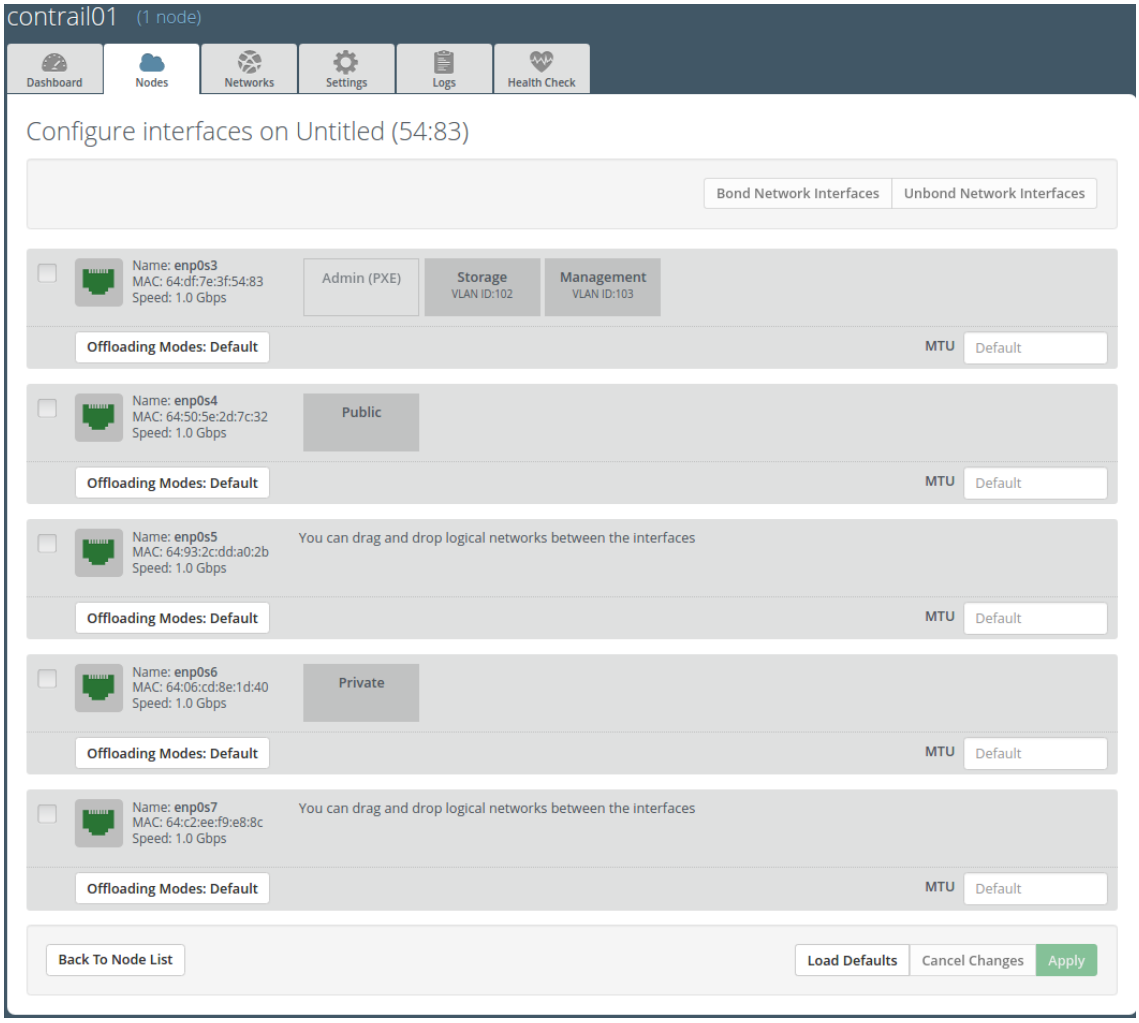


Warning: First usable addresses from the Private network will be used as VIP for Contrail controllers. For example, if your Private network CIDR is `192.168.200.0/24`, then Contrail VIP will be `192.168.200.1`. If you want to use other IP as VIP, you need to specify a range for this network.

9. Example of network configuration

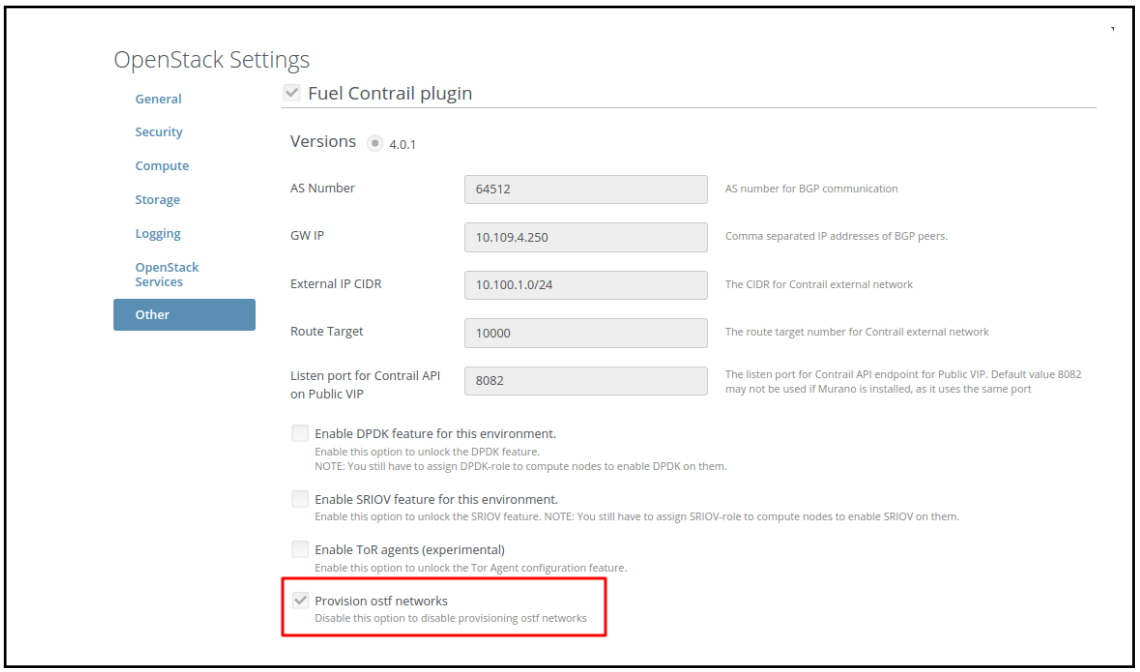
   Use hardware servers with two network interfaces as OpenStack nodes. The interfaces configuration is as follows:

   - Management and Storage networks are on the same interface with `Admin` network using tagged VLANs
   - The second interface is dedicated to Public network as untagged
   - The forth interface is dedicated to Contrail operations as untagged (Private network)



Warning:

- Be sure to launch network verification check before starting deployment. Incorrect network configuration will result in non-functioning environment.
- Some environemnts may require changes to default networks created during deployment for OSTF tests. For example, the network allocated for floating IP addresses may need some exclusions in address allocation for more specific routes. This affects the ability to deploy changes to OpenStack environments with fails on default network creation. For this reason Contrail plugin settings have an option that disables creation of default ostf networks. By default, this option is enabled.



10. Press *Deploy changes* to deploy the environment (page 25).

    After installation is finished, you can access Contrail web UI using the same IP address as OpenStack Dashboard and port 8143 through HTTPS protocol. For example, if you configured public network as described on the screenshot above, then you can access Contrail web UI through `https://<Public-VIP>:8143`.

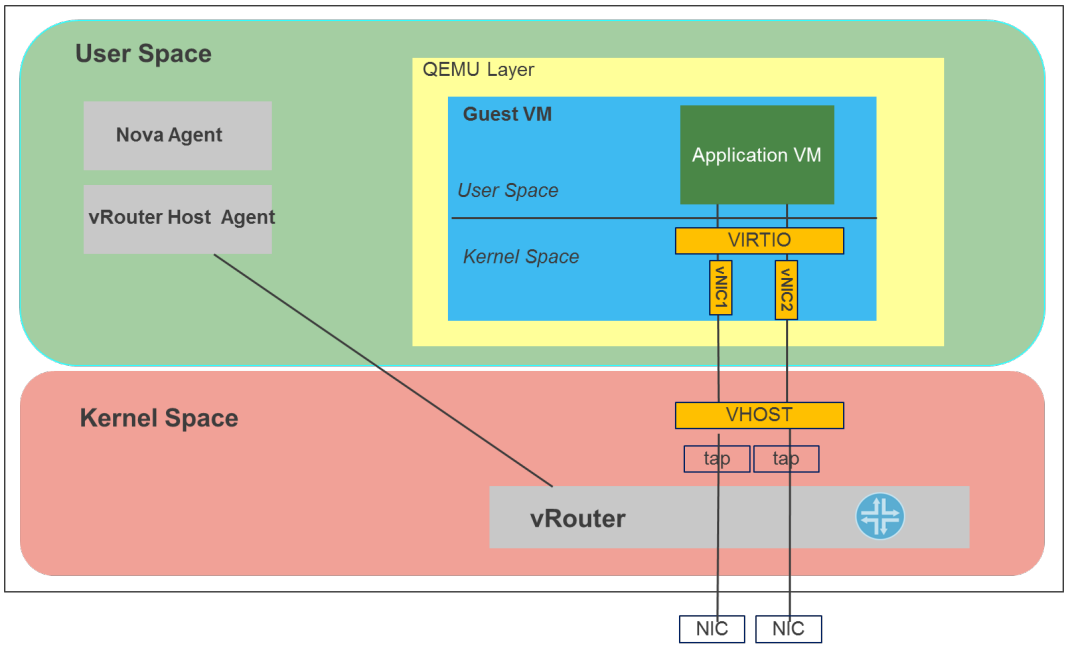
CHAPTER 6

DPDK-based vRouter

Description

The Data Plane Development Kit (DPDK) is a set of data plane libraries and network interface controller drivers for fast packet processing. The DPDK provides a programming framework for Intel x86 processors and enables faster development of high-speed data packet networking applications.

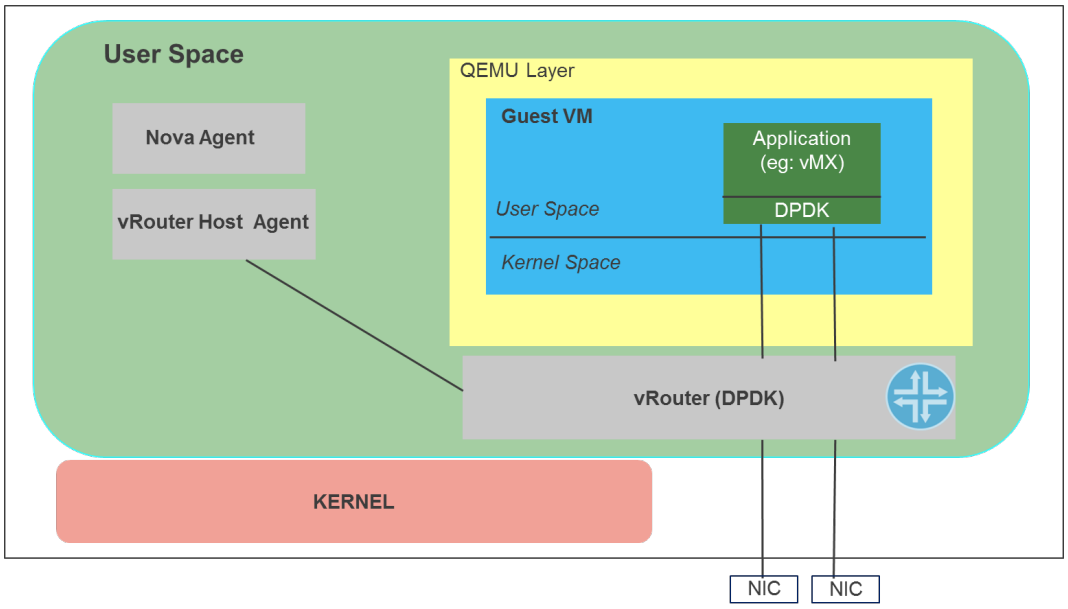
By default, Contrail virtual router (vRouter) is running as a kernel module on Linux.



The vRouter module can fill a 10G link with TCP traffic from a virtual machine (VM) on one server to a VM on another server without making any assumptions about hardware capabilities in the server NICs. Also, to support interoperability and use a standards-based approach, vRouter does not use new protocols and encapsulations. However, in network function virtualization

(NFV) scenarios, other performance metrics such as packets-per-second (pps) and latency are as important as TCP bandwidth. With a kernel module, the pps number is limited by various factors such as the number of VM exits, memory copies, and the overhead of processing interrupts.

To optimize performance for NFV use cases, vRouter can be integrated with the Intel DPDK (Data Plane Development Kit). To integrate with DPDK, vRouter can now run as a user process instead of a kernel module.



This process links with the DPDK libraries and communicates with the vRrouter host agent, which runs as a separate process. You can write an application inside of the guest VM to use the DPDK API or you can use the traditional socket API. However, for NFV applications such as vMX, which require high performance, using the DPDK API inside the VM is preferable.

Prerequisites

- Installed Fuel 8.0
- Installed Contrail plugin *Installation Guide*
- Environment must support `KVM` for compute virtualization and `Neutron with tunneling segmentation` for networking
- Network card must support DPDK. List of compatible adapters can be found on the DPDK website

Restrictions

- Only compute hosts can be configured with DPDK role. `DPDK` role is just a mark that enables DPDK feature on a certain compute node. If you try to use `DPDK` role with other roles, `DPDK` role won't have any effect.
- Contrail DPDK feature doesn't work with qemu virtualization as far as with nested KVM.

This means that for current release DPDK-based vRouter works only on baremetal computes.

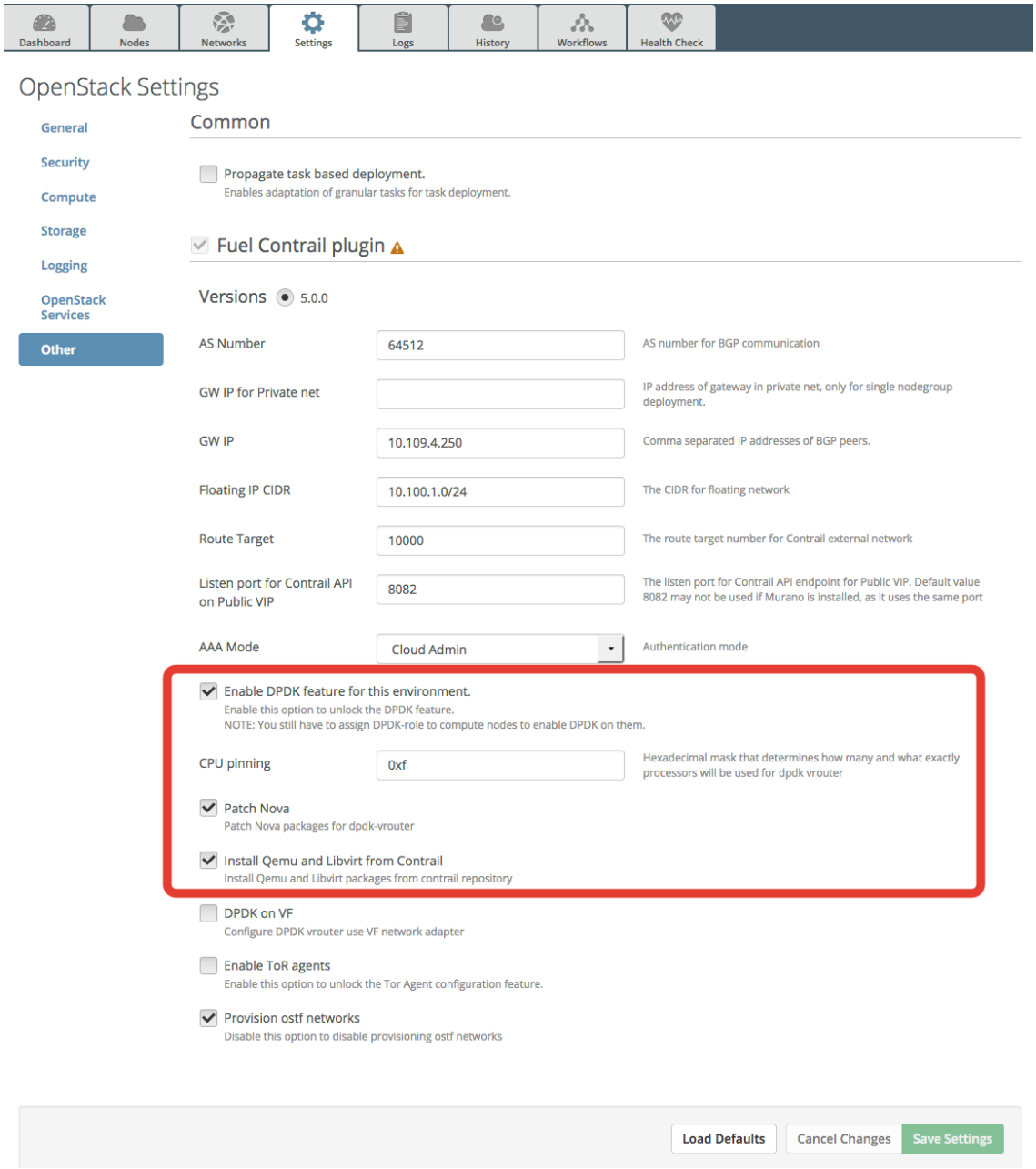
- Contrail DPDK vRouter permanently uses 1GB of hugepages. Therefore, you need to allocate enough amount of hugepages to run vRouter and VMs with DPDK.

Configure DPDK

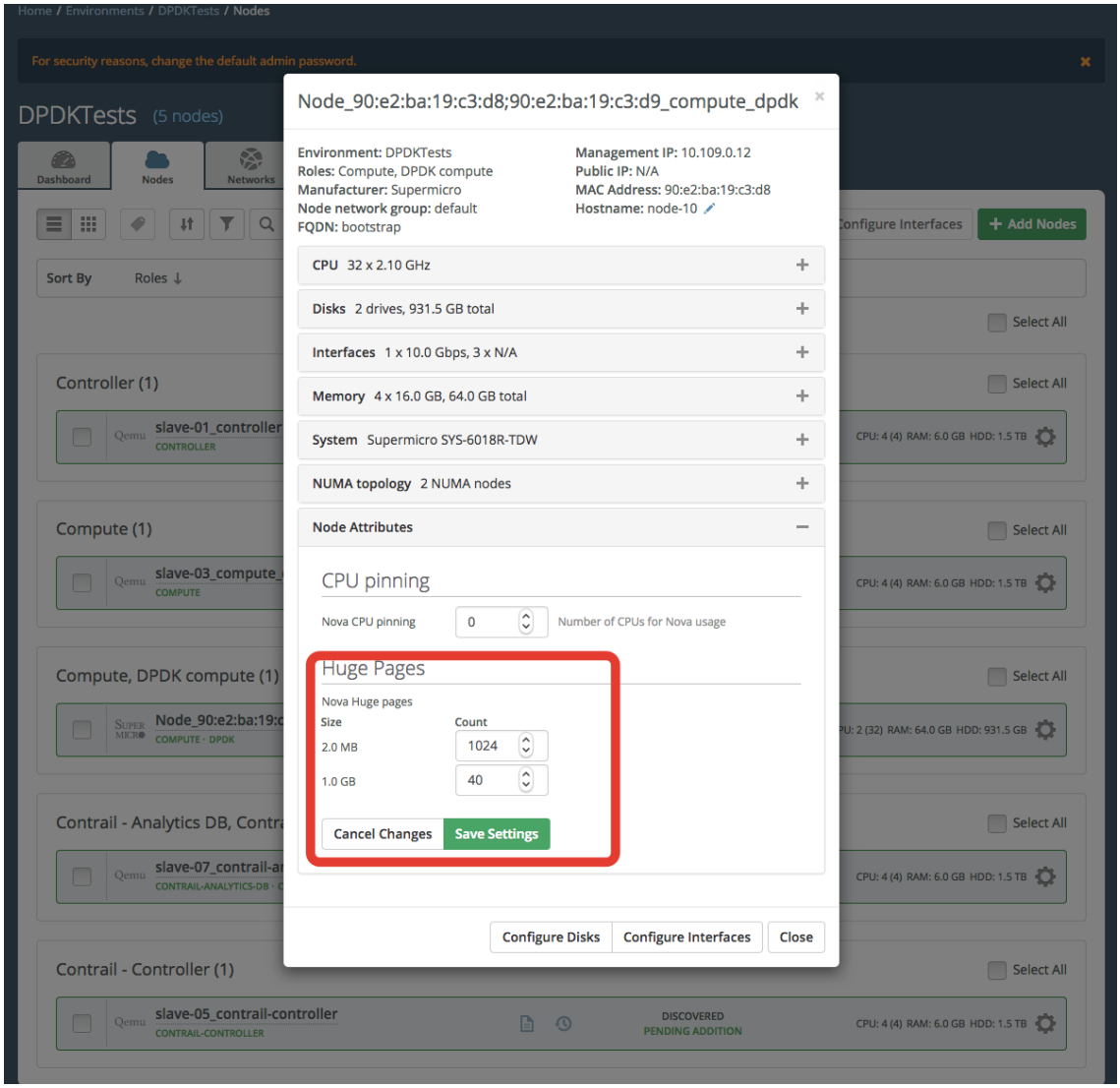
To configure DPDK you should proceed with the following steps:

1. Enable the Contrail plugin in Fuel web UI settings

2. Enable DPDK on Fuel web UI



3. Choose the size and amount of huge pages to allocate for each Compute. They will be used for both vRouter process and VMs backing. 2MB sized huge pages can be added on-fly, 1GB sized require a reboot. Also, leave some amount of memory for the operating system itself.


Home / Environments / DPDKTests / Nodes
For security reasons, change the default admin password.
DPDKTests (5 nodes)
Dashboard
Nodes
Networks
Configure Interfaces
Add Nodes
Sort By
Roles
Select All
Controller (1)
slave-01_controller
CONTROLLER
Compute (1)
slave-03_compute_
COMPUTE
Compute, DPDK compute (1)
Node_90:e2:ba:19:c
COMPUTE · DPDK
Contrail - Analytics DB, Contra
slave-07_contrail-a
CONTRAIL-ANALYTICS-DB
Contrail - Controller (1)
slave-05_contrail-controller
CONTRAIL-CONTROLLER
DISCOVERED
PENDING ADDITION
CPU: 4 (4) RAM: 6.0 GB HDD: 1.5 TB
CPU: 2 (32) RAM: 64.0 GB HDD: 931.5 GB
Node_90:e2:ba:19:c3:d8;90:e2:ba:19:c3:d9_compute_dpdk
Environment: DPDKTests
Roles: Compute, DPDK compute
Manufacturer: Supermicro
Node network group: default
FQDN: bootstrap
Management IP: 10.109.0.12
Public IP: N/A
MAC Address: 90:e2:ba:19:c3:d8
Hostname: node-10
CPU 32 x 2.10 GHz
Disks 2 drives, 931.5 GB total
Interfaces 1 x 10.0 Gbps, 3 x N/A
Memory 4 x 16.0 GB, 64.0 GB total
System Supermicro SYS-6018R-TDW
NUMA topology 2 NUMA nodes
Node Attributes
CPU pinning
Nova CPU pinning
0
Number of CPUs for Nova usage
Huge Pages
Nova Huge pages
Size
Count
2.0 MB
1024
1.0 GB
40
Cancel Changes
Save Settings
Configure Disks
Configure Interfaces
Close

4. Add `DPDK` role on computes where you want to have DPDK-based vRouter.

Note: Computes that are not marked with DPDK role will use kernel-based vRouter.

5. Deploy environment

Warning: Computes with DPDK-based vRouter require flavor with Huge Pages enabled. Instances with usual flavours can't be launched on DPDK-enabled hosts.

If DPDK is enabled in plugin settings, Fuel will create one flavor that will have Huge Pages support, named `m1.small.hpgs`. To create a custom flavor, follow the steps below on the controller node:

```
# . openrc
# nova flavor-create m2.small.hpgs auto 2000 20 2
# nova flavor-key m2.small.hpgs set hw:mem_page_size=large
# nova flavor-key m2.small.hpgs set aggregate_instance_extra_specs:hpgs=true
```

Verify DPDK

To verify your installation, proceed with basic checks below:

1. Verify that Contrail services and DPDK vRouter are running on a compute node:

```
contrail-status
```

System response:

```
== Contrail vRouter ==
supervisor-vrouter:           active
contrail-vrouter-agent        active
contrail-vrouter-dpdk         active
contrail-vrouter-nodemgr      active
```

2. Verify if DPDK vRouter binds network interfaces:

```
/opt/contrail/bin/dpdk_nic_bind.py -s
```

Example of system response:

```
Network devices using DPDK-compatible driver
============================================
0000:06:00.0 '82599ES 10-Gigabit SFI/SFP+ Network Connection' drv=igb_uio
unused=
Network devices using kernel driver
===================================
0000:02:00.0 'I350 Gigabit Network Connection' if=eth0 drv=igb unused=igb_uio
0000:02:00.1 'I350 Gigabit Network Connection' if=eth1 drv=igb unused=igb_uio
0000:06:00.1 '82599ES 10-Gigabit SFI/SFP+ Network Connection' if=eth3 drv=ixgbe
unused=igb_uio
Other network devices
=====================
<none>
```

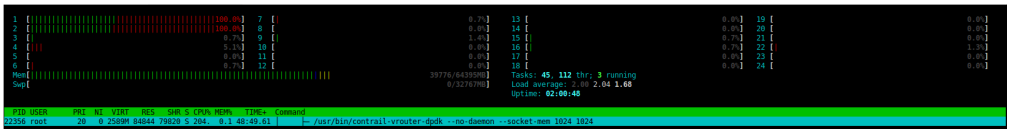
3. Verify if vRrouter uses Huge Pages:

```
grep Huge /proc/meminfo
```

Example of system response:

```
AnonHugePages:         0 kB
HugePages_Total:   30000
HugePages_Free:    29488
HugePages_Rsvd:        0
HugePages_Surp:        0
Hugepagesize:       2048 kB
```

4. Verify if vRouter uses CPU:



5. Verify if vRouter creates interface after creation of a virtual machine:

```
vif --list
```

Example of system response:

```
Vrouter Interface Table
Flags: P=Policy, X=Cross Connect, S=Service Chain, Mr=Receive Mirror
       Mt=Transmit Mirror, Tc=Transmit Checksum Offload, L3=Layer 3, L2=Layer 2
       D=DHCP, Vp=Vhost Physical, Pr=Promiscuous, Vnt=Native Vlan Tagged
       Mnp=No MAC Proxy, Dpdk=DPDK PMD Interface, Rfl=Receive Filtering Offload
       Mon=Interface is Monitored, Uuf=Unknown Unicast Flood
       Vof=VLAN insert/strip offload
vif0/0      PCI: 0:0:0.0 (Speed 10000, Duplex 1)
            Type:Physical HWaddr:00:1b:21:87:21:98 IPaddr:0
            Vrf:0 Flags:L3L2Vp MTU:1514 Ref:14
            RX device packets:3671  bytes:513937 errors:10
            RX port   packets:3671 errors:0
            RX queue  packets:6 errors:0
            RX queue errors to lcore 0 0 0 0 0 0 0 0 0 0 0 0
            RX packets:3671  bytes:499253 errors:0
            TX packets:4049  bytes:2135246 errors:0
            TX port   packets:4049 errors:0
            TX device packets:4049  bytes:2135246 errors:0
vif0/1      Virtual: vhost0
            Type:Host HWaddr:00:1b:21:87:21:98 IPaddr:0
            Vrf:0 Flags:L3L2 MTU:1514 Ref:8
            RX port   packets:4111 errors:0
            RX queue  packets:4093 errors:0
            RX queue errors to lcore 0 0 0 0 0 0 0 0 0 0 0 0
            RX packets:4111  bytes:2143597 errors:0
            TX packets:3786  bytes:509223 errors:0
            TX queue  packets:790 errors:0
            TX port   packets:3771 errors:0
vif0/2      Socket: unix
            Type:Agent HWaddr:00:00:5e:00:01:00 IPaddr:0
            Vrf:65535 Flags:L3 MTU:1514 Ref:2
            RX port   packets:45 errors:0
            RX queue errors to lcore 0 0 0 0 0 0 0 0 0 0 0 0
            RX packets:45  bytes:4322 errors:3
            TX packets:951  bytes:95940 errors:0
            TX queue  packets:951 errors:0
            TX port   packets:951 errors:0 syscalls:951
vif0/3      Ethernet: veth1404577d-b
            Type:Virtual HWaddr:00:00:5e:00:01:00 IPaddr:0
            Vrf:2 Flags:PL3L2D MTU:9160 Ref:11
            RX port   packets:31 errors:0
            RX queue  packets:24 errors:0
            RX queue errors to lcore 0 0 0 0 0 0 0 0 0 0 0 0
            RX packets:31  bytes:18164 errors:0
            TX packets:19  bytes:1091 errors:4
            TX queue  packets:14 errors:0
            TX port   packets:15 errors:0
```

Change DPDK options

This chapter describes how to change DPDK related options from Fuel web UI:

- *Enable DPDK feature for this environment* - this option enables DPDK globally. Still you must use `DPDK` role to mark a compute node with DPDK
- *Hugepage size* - specifies the size of huge pages that will be used for a dpdk feature. Verify if 1GB pages are supported on the target compute node:

```
grep pdpe1gb /proc/cpuinfo | uniq
```

- *Hugepages amount (%)* - sets amount of memory allocated on each compute node for huge pages. It will use `%` of all memory available on a compute node. DPDK vRouter permanently uses 1GB of huge pages and other applications running on compute node may not support huge pages. Therefore, use this parameter carefully.
- *CPU pinning* - this hexadecimal value describes how many and which exact processors `dpdk-vrouter` will use. CPU pinning is implemented using taskset util
- *Patch Nova* - in the MOS 8.0 release nova doesn't support DPDK-based vRouter. In future, MOS maintenance updates will include necessary patches.
- *Install Qemu and Libvirt from Contrail* - DPDK-based vRouter needs huge pages memory-backing for guests. MOS 8.0 includes qemu and libvirt that don't support huge pages memory-backing. DPDK feature needs qemu and libvirt from Contrail only on nodes with `DPDK` role.

Change Huge Pages settings after deployment

After deploy is finished, plugin settigs are locked in Fuel web UI. Therefore, size and ammount of huge pages cannot be changed by the plugin. You need to set Huge Pages settings manually on each compute node.

To set 2MB-sized huge pages:

1. Calculate the number of huge pages based on the ammount you need. For example 20GB = 20 * 1024 / 2 = 10240 pages.
2. Set 2MB-sized huge pages:

```
sysctl -w vm.nr_hugepages=<number of pages>
```

3. Edit the `/etc/sysctl.conf` file to make these changes persistent over reboots.

On the contrary to setting 2MB-sized huge pages, you can set 1GB-sized huge pages through the kernel parameter only, which requires a reboot to take effect. Kernel versions supplied with Ubuntu 14.04 don't support on the fly allocation for 1GB-sized huge pages.

To set 1GB-sized huge pages:

1. Edit the `/etc/default/grub` file and set needed amount of huge pages. For GRUB_CMDLINE_LINUX in `/etc/default/grub`:

```
GRUB_CMDLINE_LINUX="$GRUB_CMDLINE_LINUX hugepagesz=1024M hugepages=20
```

2. Update the bootloader and reboot for these parameters to take effect:

```
# update-grub
# reboot
```

CHAPTER 7

DPDK-based vRouter on virtual function (VF)

Description

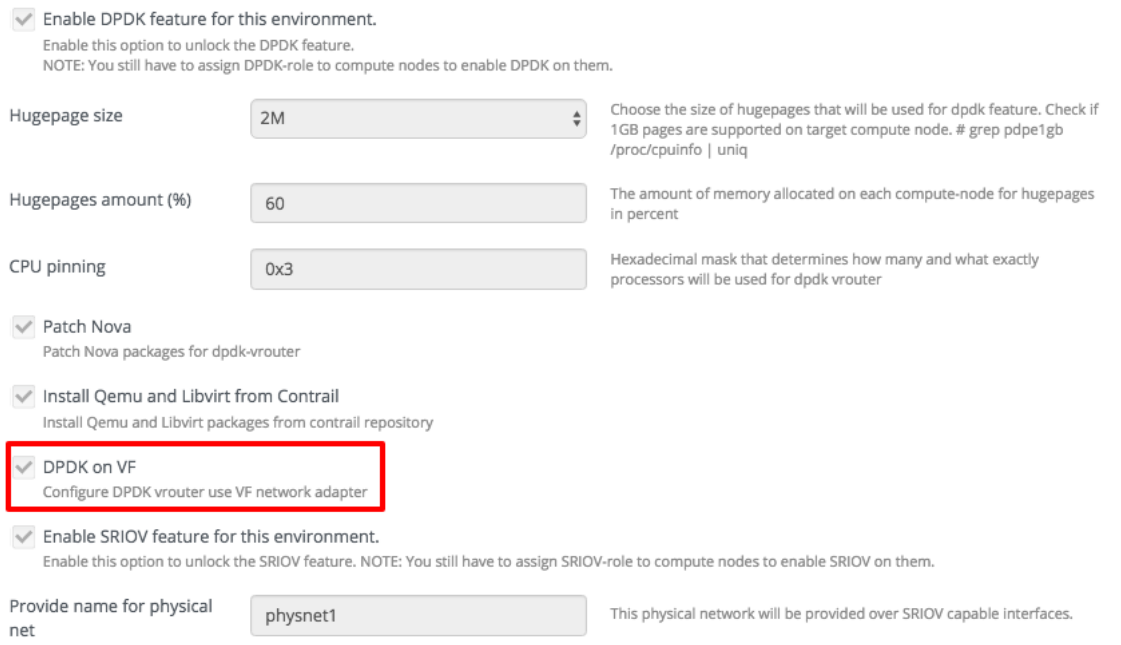
This guide describes how to run DPDK-based vRouter on virtual functions (VF). DPDK on VF depends on *Enable SR-IOV* and *DPDK-based vRouter* features. This feature shares a physical interface for DPDK and SR-IOV usage.

Prerequisites

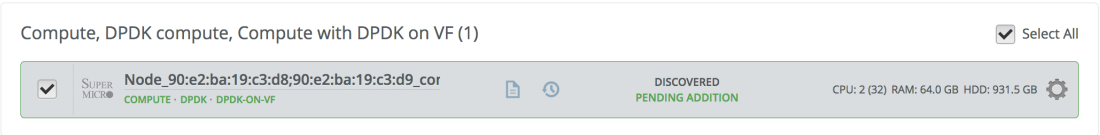
- Installed Fuel 9.1
- Installed Fuel Contrail Plugin *Installation Guide*
- Environment must be created with "KVM" for compute virtualization and "Contrail" for networking
- Network card must support DPDK. List of compatible adapters can be found on DPDK website
- Network card must support SRIOV.

How to enable DPDK on VF

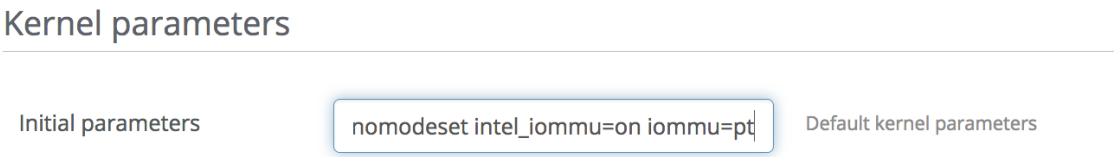
1. Enable DPDK feature *DPDK-based vRouter*.
2. Enable DPDK on VF in Fuel UI settings:



3. Assign `Compute`, `DPDK`, and `DPDK-on-VF` roles to the host where you want to enable DPDK on VF feature:



4. Add `intel_iommu=on iommu=pt` to kernel parameters:



5. Deploy environment

If DPDK on VF is enabled in plugin settings, it will be deployed on computes with `DPDK` and `DPDK-on-VF` roles. During deploy following configurations will be made on compute nodes with DPDK and SR-IOV roles:

1. Virtual functions will be allocated on private interface.
2. First VF will be used for DPDK-based vRouter.
3. Rest of the VFs will be added to `pci_passthrough_whitelist` setting in `nova.conf` for SR-IOV usage.

CHAPTER 8

Enable SR-IOV

Prerequisites

This guide assumes that you have installed Fuel and performed steps 5.3.1 - 5.3.9 from *Installation Guide*. To enable SR-IOV, you need a SRIOV-capable network PCI card. Also, only compute hosts can be configured with the‘‘SRIOV‘‘ role.

Features

1. You can have multple VLANs inside one physical network
2. When using `Passthrough`, as in SR-IOV scenario, OpenStack does not provides dhcp and metadata. You have to manage that manually or provide an additional network port with a regular OpenStack network.

SR-IOV Description

Quoting Mirantis blog post:

SR-IOV is a PCI Special Interest Group (PCI-SIG) specification for virtualizing network interfaces, representing each physical resource as a configurable entity (called a PF for Physical Function), and creating multiple virtual interfaces (VFs or Virtual Functions) with limited configurability on top of it, recruiting support for doing so from the system BIOS, and conventionally, also from the host OS or hypervisor. Among other benefits, SR-IOV makes it possible to run a very large number of network-traffic-handling VMs per compute without increasing the number of physical NICs/ports and provides means for pushing processing for this down into the hardware layer, off-loading the hypervisor and significantly improving both throughput and deterministic network performance.

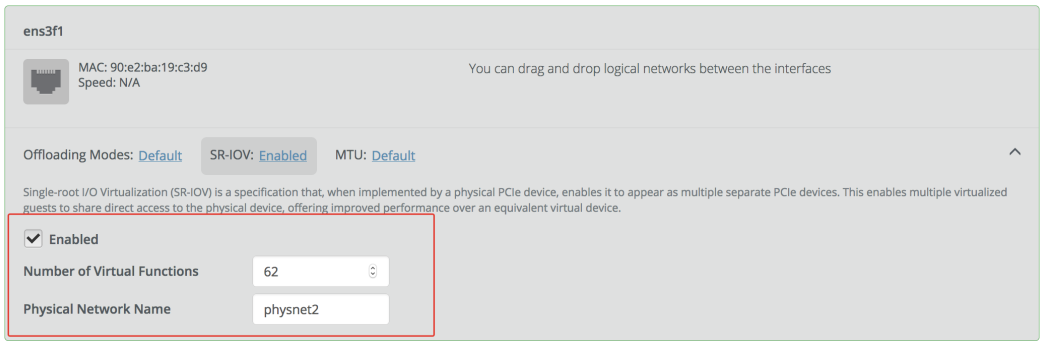
Verify SR-IOV environment

To verify if network interface is SRIOV-capable and how many VFs are available, run the following command on the boostraped host:

```
lspci -s <bus ID> -vvv
```

Enable SR-IOV in Fuel

To enable SR-IOV in Fuel go to node interface configuration and enable it as shown on picture



1. Deploy as in 5.3.10 *Installation Guide*

Create a virtual machine with SR-IOV device

To create a virtual machine with SR-IOV device:

1. Create a VM with configured physical network and VLAN id:

```
neutron net-create \
--provider:physical_network=<physical network from contrail settings tab> \
--provider: segmentation_id=<Vlan_id> <Network_Name>
```

2. Create a subnet:

```
neutron subnet-create <Network_name> <Subnet>
```

3. Create a port:

```
neutron port-create \
--fixed-ip subnet_id=<subnet uuid>,ip_address=<IP address from above subnet> \
--name <name of port> <vn uuid> --binding:vnic_type direct
```

4. Boot the VM with the port:

```
nova boot \
--flavor m1.large --image <image name> \
--nic port-id=<uuid of above port> <vm name>
```

CHAPTER 9

Contrail TSN

TSN Description

Contrail supports extending a cluster to include bare metal servers and other virtual instances connected to a TOR switch supporting OVSDB protocol. You can configure the bare metal servers and other virtual instances to be a part of any of the virtual networks configured in the contrail cluster facilitating communication between them and the virtual instances running in the cluster. You can use Contrail policy configurations to control this communication.

The solution uses the OVSDB protocol to configure the TOR switch and to import dynamically learnt addresses from it. VXLAN encapsulation will be used in the data plane communication with the TOR switch.

A new node, the TOR services node (TSN), is introduced and provisioned as a new role in the Contrail system. The TSN acts as the multicast controller for the TOR switches. TSN also provides DHCP and DNS services to the bare metal servers or virtual instances running behind TOR ports.

TSN receives all the broadcast packets from the TOR, and replicates them to the required compute nodes in the cluster and to other EVPN nodes. Broadcast packets from the virtual machines in the cluster are sent directly from the respective compute nodes to the TOR switch.

TSN can also act as the DHCP server for the bare metal servers or virtual instances, leasing IP addresses to them, along with other DHCP options configured in the system. TSN also provides a DNS service for the bare metal servers.

See also:

Contrail Wiki

Prerequisites

This guide assumes that you have installed Fuel and all the nodes of your future environment are discovered and functional. To configure TSN in you environment, you need to perform steps

additional to *Installation Guide*

To configure TSN in your network, you need TOR switch.

HA implementation details

Contrail TSN in HA mode is implemented for default SSL mode.

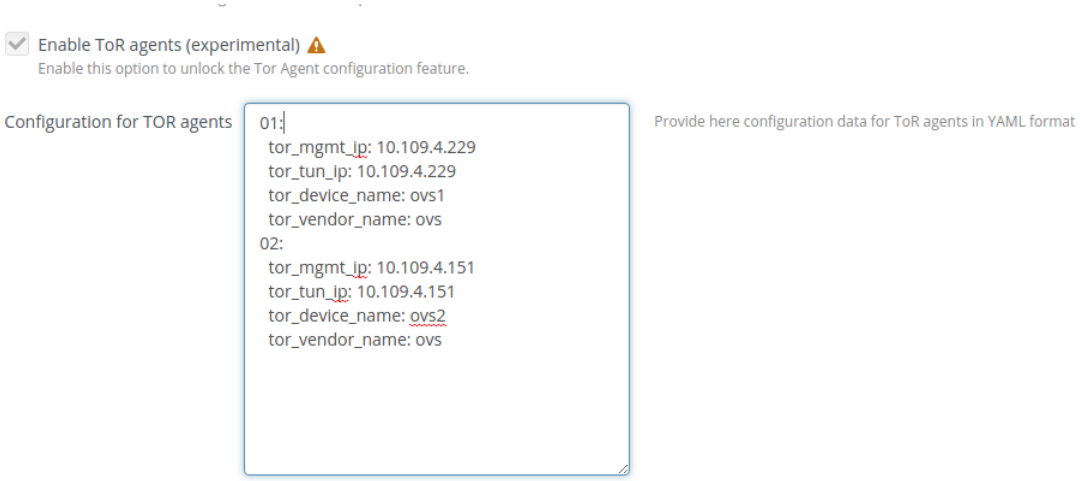
All required certificates will be located on CONTRAIL-TSN node in location:

```
/var/lib/astute/tsn_certificates/certs
```

There will be two folders per TOR service. Folder named tor_agent_<@id> contain certs for tor agent service. Folder named vtep_<@id> contain certificates which should be delivered to ToR Switch.

Configure TSN

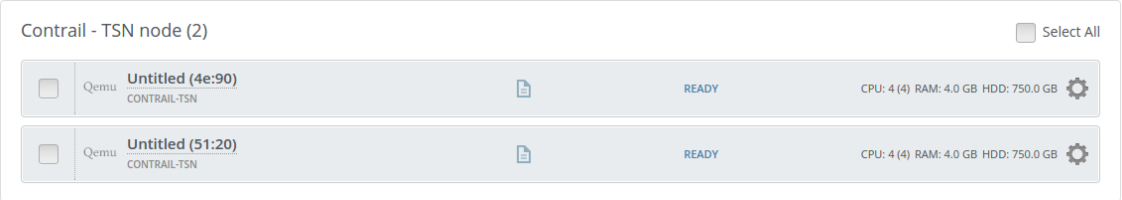
1. Enable ToR Agents



2. Provide Tor Agents configuration in YAML format, based on example

```
01:
  tor_mgmt_ip: 10.109.4.150
  tor_tun_ip: 10.109.4.150
  tor_device_name: ovs1
  tor_vendor_name: ovs
02:
  tor_mgmt_ip: 10.109.4.151
  tor_tun_ip: 10.109.4.151
  tor_device_name: ovs2
  tor_vendor_name: ovs
```

3. Deploy additional node/nodes with CONTRAIL-TSN role



4. Configure ToR Switches with SSL certificates located on TSN node in:

```
/var/lib/astute/tsn_certificates/certs
```

5. Verify working TSN by going to Contrail web UI

CHAPTER 10

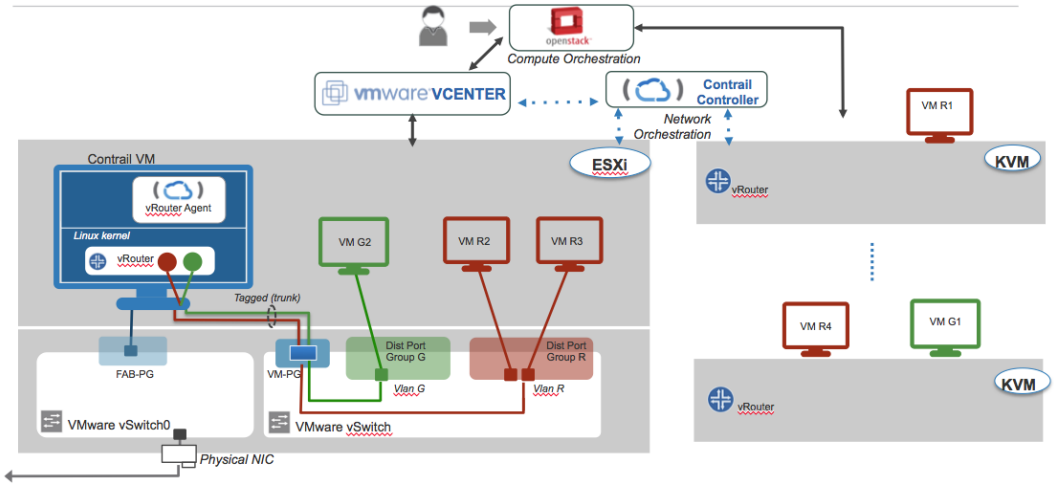
Contrail with VMware vCenter

Overview

Starting from Contrail Release 3.0.0, it is possible to integrate Contrail with the VMware vCenter acting as an Openstack compute node.

This topic describes how configure Fuel Contrail Plugin to work with existing or already provisioned vSphere deployments that use OpenStack as the main orchestrator.

Integration with vCenter include two main roles: compute-vmware and contrail-vmware. As the basis for compute-vmware role will use default Fuel compute-vmware role. Compute-vmware will be located on the openstack side of hybrid environment and will include nova-compute with Contrail Nova vCenter driver. One compute-vmware will serve one vCenter. In the current release work with multiple vCenter instances is not supported. Compute-vmware role will be not compatible with any other role. Contrail-vmware will be located on vmware side of hybrid environment and will include Contrail vRouter. One contrail-vmware must to be installed on each ESXi node. Contrail-vmware role will not be compatible with any other role. Integration assumes that vmware part of the environment already exists - datacenter and clusters are created. Deployment of the environment will include 2 stages. During the 1st stage user will run script that prepares vmware part for deployment (creates few Distributed Switches and spawns virtual machine on each ESXi node). The rest of management will provided by the Fuel master



Prerequisites

- Installed Fuel 9.1
- Installed Fuel Contrail Plugin *Installation Guide*
- Environment must be created with "vCenter" support for compute virtualization and Contrail for networking
- vSphere environments must be already preconfigured
- pyvmomi python package need to be installed for vmware provision script

Restrictions

- There must be a single vmware-compute node for each vCenter
- Compute-vmware role can not be combined with any other roles
- Contrail-vmware role can not be combined with any other roles
- The environment must contain at least one KVM/QEMU compute node
- According contrail package (version: 3.1.0.0-25) bug we can use only one cluster per Datacenter.
- All ESXi hosts in vCenter cluster must have instance with contrail-vmware role

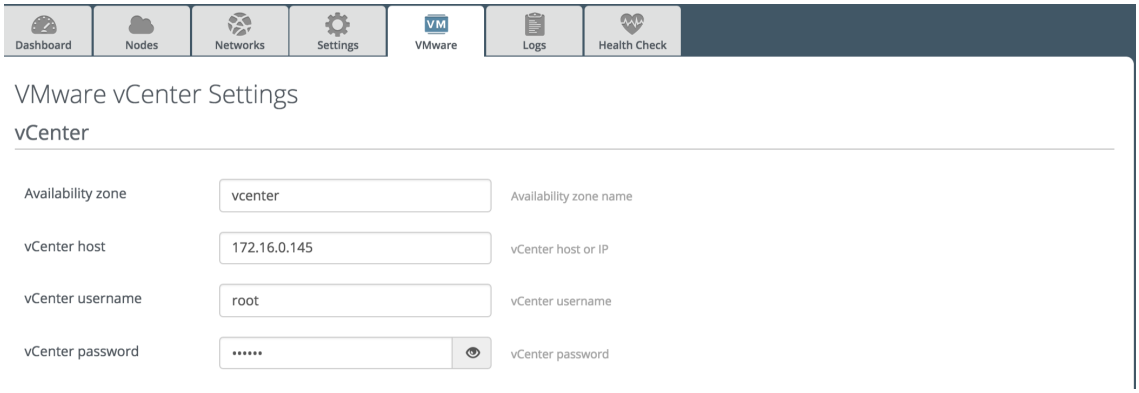
Configuration

To install environment with Contrail and VMWware support you should proceed with following steps:

1. Install pyvmomi module

```
[root@nailgun ~]# easy_install pyvmomi
```

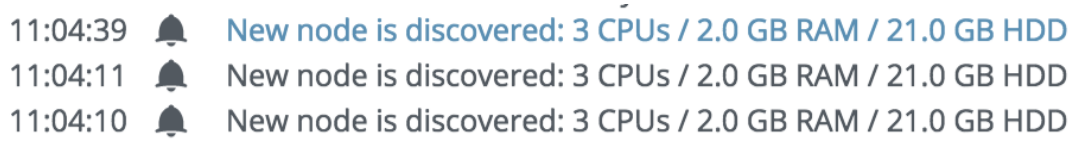
2. Fill vmware credentials in Fuel vmware tab



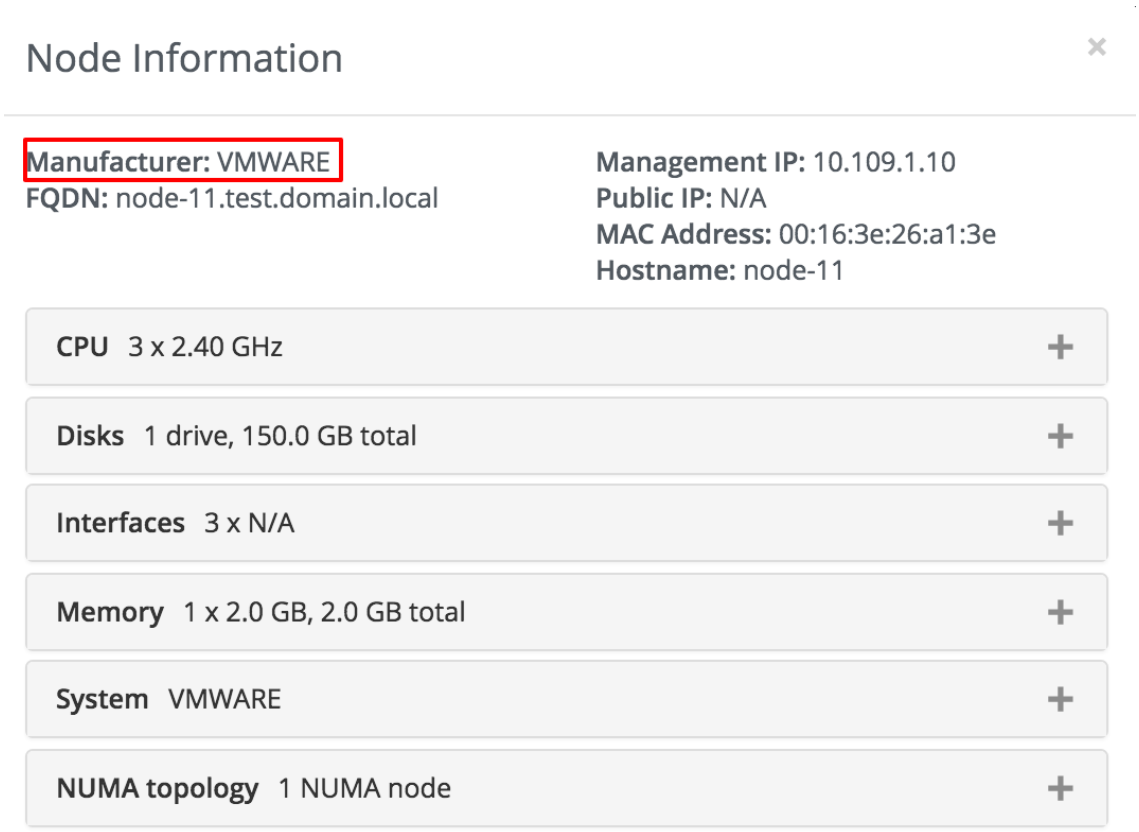
3. Run script that will spawn ContrailVM's, DVS's

```
[root@nailgun ~]# cd /var/www/nailgun/plugins/contrail-5.1/deployment_
↪scripts/
[root@nailgun deployment_scripts]# ./spawner.py --env_id 1 --spawn
```

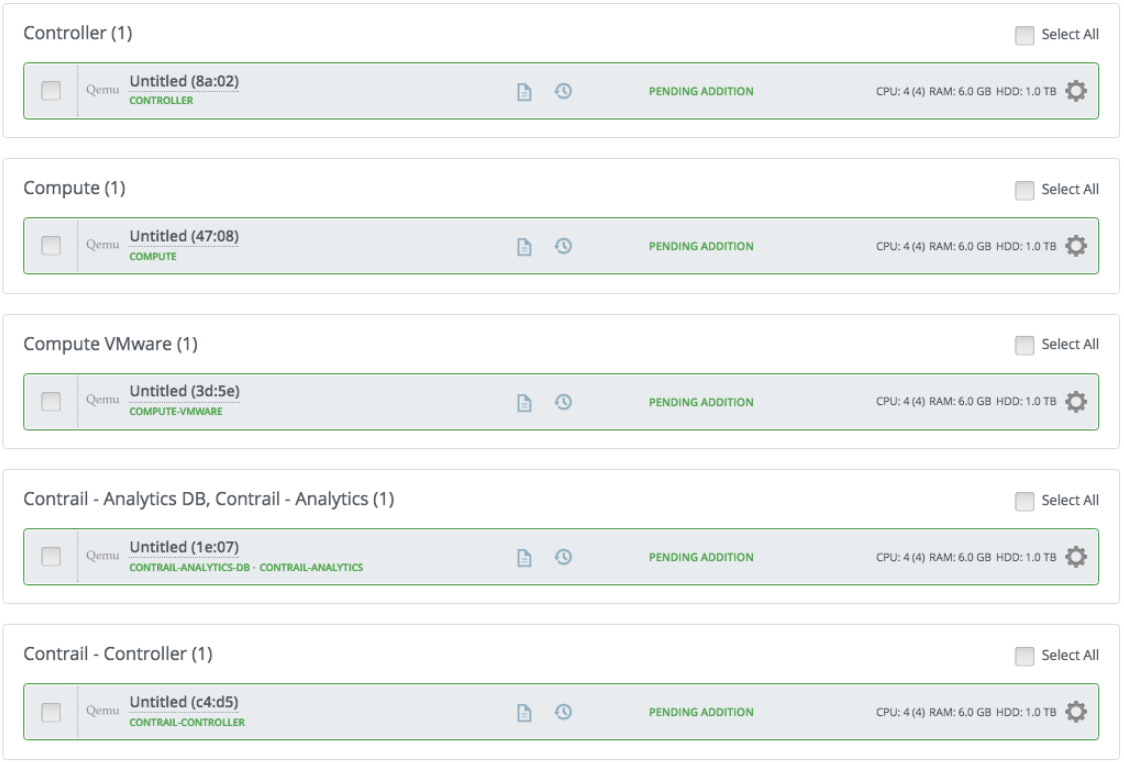
4. Wait a few minutes when ContrailVM's node will be arrived



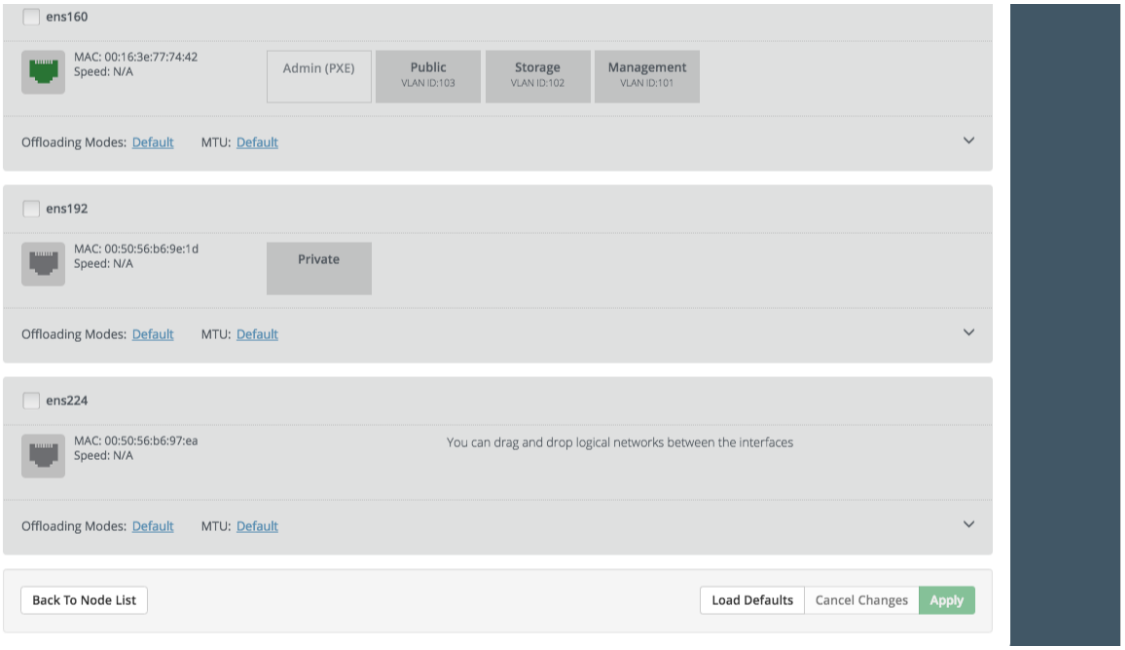
5. To verified if the nodes from vmware you may check "Node Information"



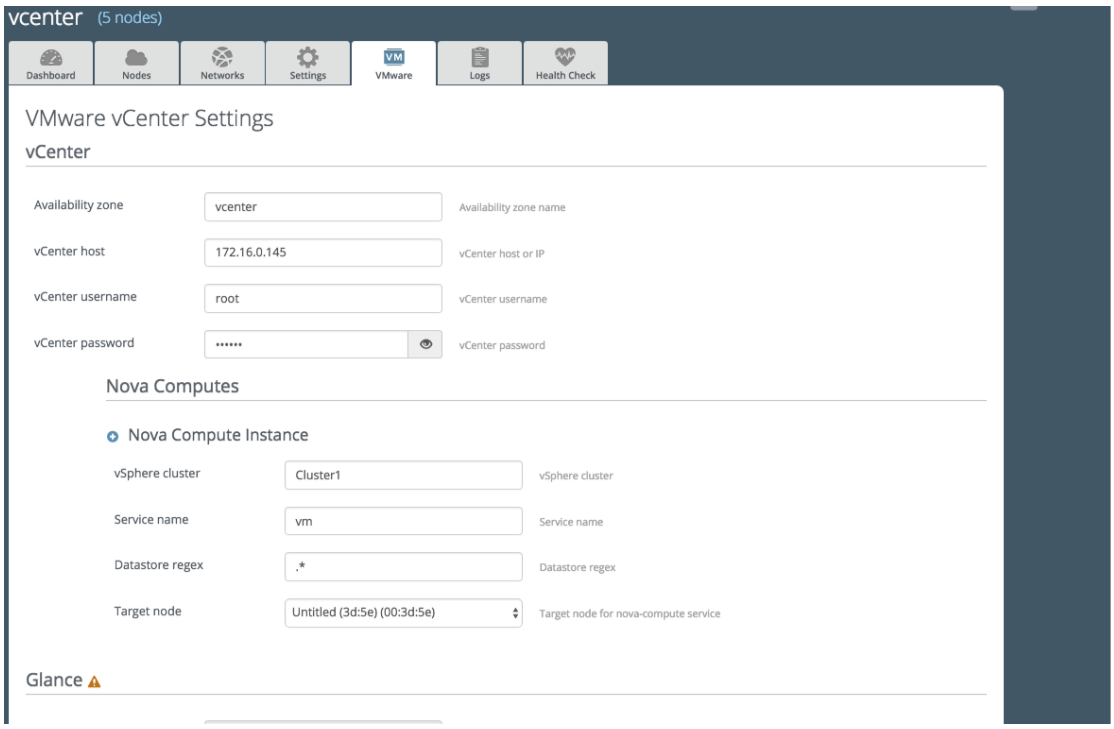
6. Assign all planned roles (including **single compute-vmware** role and contrail-vmware for each ESXi host) in Nodes tab from Fuel UI



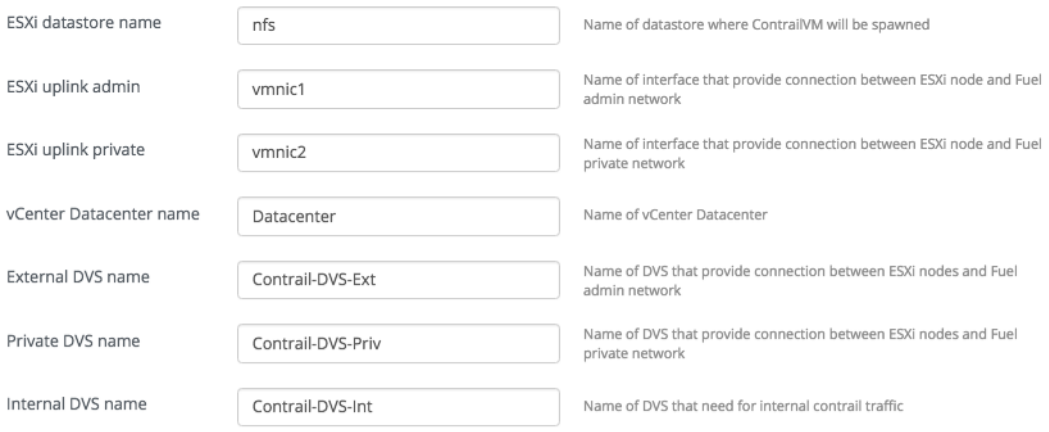
7. ContrailVM's will be spawned with 3 network interfaces (the first two for fuel networks and third for vmware connection) therefore we assume that public, storage and management network will use vlan tagging



8. Fill settings in VMware tab from Fuel UI



9. Fill additional settings in Fuel Contrail plugin settings from Fuel UI



10. Deploy environment

Verification

After deploy finishes, you can verify your installation.

1. Check that Contrail services are running on compute node:

```
root@node-35:~# contrail-status
== Contrail vRouter ==
supervisor-vrouter:           active
contrail-vrouter-agent        active
contrail-vrouter-nodemgr      active
```

2. Check that Contrail services are running on ContrailVM:

```
root@ContrailVM-249:~# contrail-status
== Contrail vRouter ==
supervisor-vrouter:           active
contrail-vrouter-agent        active
contrail-vrouter-nodemgr      active
```

VMware related options

From VMware tab:

- *Availability zone* - openstack availability zone name
- *vCenter host* - vCenter host or IP
- *vCenter username* - vCenter username
- *vCenter password* - vCenter password
- *vSphere cluster* - name of vSphere cluster
- *Service name* - nova-compute service name on compute-vmware
- *Datastore regex* - Datastore regex
- *Target node* - Target node for nova-compute service

From Fuel Contrail plugin settings:

- *ESXi datastore name* - Name of datastore where ContrailVM will be spawned
- *ESXi uplink admin* - Name of interface that provide connection between ESXi node and Fuel admin network
- *ESXi uplink private* - Name of interface that provide connection between ESXi node and Fuel private network
- *vCenter Datacenter name* - name of vCenter Datacenter
- *External DVS name* - Name of DVS that provide connection between ESXi and Fuel admin network
- *Private DVS name* - Name of DVS that provide connection between ESXi and Fuel private network
- *Internal DVS name* - Name of DVS need for internal contrail traffic

spawer.py parameter description:

- *–env_id*(type int) - id of Fuel environment (mandatory parameter)
- *–spawn*(type boolean) - spawn vm's for contrail-vmware role. When run script with this parameter it will take credential from Fuel vmware tab, create dvs's with port groups, spawn vm's on each ESXi host in cluster and attach their to all dvs's.
- *–map-ips*(type boolean) - this need for internal plugin calculations
- *–dvs-mtu-ext*(type int) - set max MTU for external DVS
- *–dvs-mtu-priv*(type int) - set max MTU for private DVS
- *–dvs-mtu-int*(type int) - set max MTU for internal DVS

- *–cluster-list*(type str) - change cluster list in Fuel. This option is deprecated no need to you it.
- *–reduce-vm-params*(type boolean) - Reduce memory value for ContrailVM's, for production purposes this parameter not recommend to use.

Add and delete ESXi hosts

Add ESXi host:

1. Add ESXi host to vCenter cluster manually
2. Run script that will spawn additional ContrailVM

```
[root@nailgun ~]# cd /var/www/nailgun/plugins/contrail-5.1/deployment_
↪scripts/
[root@nailgun deployment_scripts]# ./spawner.py --env_id 1 --spawn
```

3. Wait a few minutes when ContrailVM's node will be arrived
4. Assign contrail-vmware role on new ContrailVM
5. Run "Deploy Changes"

Remove ESXi host:

1. In Fuel UI remove contrail-vmware instance that located on ESXi host which you want to remove.
2. Run "Deploy Changes"
3. Remove ESXi host from vCenter cluster manually

CHAPTER 11

Contrail upgrades (experimental)

Description

Starting from version 4.0.1 the Fuel Contrail Plugin includes the set of tasks and scripts that allow the cloud administrator to upgrade the Contrail packages along with Contrail configuration with minimal downtime to production network. The upgrade process is divided into tasks, that modify only the components that need to be upgraded without touching other OpenStack components. The packages are updated using the plugin-based repository, and configuration files are updated using the templates included in the latest plugin version. Controllers and compute nodes are upgraded separately, using puppet manifests provided with plugin. Other contrail-specific roles such as DPDK-compute, VMWare-compute, and TSN are not supported yet. More information on running custom deployment graphs can be found in Fuel User Guide

Prequisites

This guide assumes that you have installed Fuel 9.1 with the Fuel Contrail plugin, and successfully deployed the environment according to *Installation Guide*.

Package versions supported:

- Fuel Contrail plugin >= 5.0.0
- Juniper Contrail >= 3.1.0

Update the packages on Fuel Master node

In case Fuel Contrail plugin package should be upgraded, please execute steps 1-3, otherwise proceed to step 4.

1. Obtain the latest package of Fuel Contrail plugin that supports your Fuel version.
2. Copy the rpm package downloaded at previous step to the Fuel Master node

```
scp contrail-5.1-5.1.0-1.noarch.rpm <Fuel Master node ip>:/tmp/
```

3. Log in to the Fuel Master node and upgrade the plugin:

```
ssh <the Fuel Master node ip>
fuel plugins --update /tmp/contrail-5.1-5.1.0-1.noarch.rpm
```

4. Copy the latest Juniper Contrail installation package to the Fuel Master node and run the installation script to unpack the vendor package and populate the plugin repository with up-to-date packages:

```
scp contrail-install-packages_3.2.0.0-19~mitaka_all.deb \
    <Fuel Master node ip>:/var/www/nailgun/plugins/contrail-5.1/
ssh <Fuel Master node ip> /var/www/nailgun/plugins/contrail-5.1/install.sh
```

Upgrade Contrail and OpenStack Controllers

The first upgrade step involves the controllers, both for OpenStack and Contrail. Upgrade tasks stop Contrail config services for the time of upgrade, this will stop Neutron operations for 10-20 minutes without affecting the workload. The Contrail control nodes will be upgraded and restarted one-by-one to keep BGP and XMPP connectivity. After the tasks have been finished on contrail nodes, the upgrade of OpenStack controllers starts. The Neutron service will be restarted in case if contrail core plugin will be upgraded.

1. Log in to Fuel Master node, change the working directory to plugin folder:

```
ssh <the Fuel Master node ip>
cd /var/www/nailgun/plugins/contrail-5.1/
```

2. Check ID of contrail plugin:

```
fuel plugins
```

3. Check ID of your env:

```
fuel2 env list
```

4. Upload upgrade graph:

```
fuel2 graph upload --plugin <plugin-ID> --type contrail_upgrade_control --file␣
↪upgrade_control.yaml
```

5. Verify the graph has been uploaded:

```
fuel2 graph list --env <env-ID>
```

6. Execute the custom graph to upgrade control plane:

```
fuel2 graph execute --env <env-ID> --type contrail_upgrade_control
```

7. Run the contrail service verification steps from *Verify Contrail plugin* to ensure that all Contrail services are up and running. You can verify the version of Contrail packages using Contrail Web UI or `contrail-version` CLI command.

Upgrade Compute nodes

After the control plane has been upgraded, you can upgrade OpenStack Compute nodes. The upgrade task can install the latest version of Contrail vRouter, correctly replacing the kernel module without host reboot. The task upgrades compute hosts one by one, in ascending order by node ID. The instances running on particular compute node will lose network connectivity during the vRouter upgrade, this can take up to 5 min.

1. Log in to Fuel Master node, change the working directory to plugin folder:

```
ssh <the Fuel Master node ip>
cd /var/www/nailgun/plugins/contrail-5.1/
```

2. Check ID of contrail plugin:

```
fuel plugins
```

3. Check ID of your env:

```
fuel2 env list
```

4. Upload upgrade graph:

```
fuel2 graph upload --plugin <plugin-ID> --type contrail_upgrade_compute --file␣
↪upgrade_compute.yaml
```

5. Verify the graph has been uploaded:

```
fuel2 graph list --env <env-ID>
```

6. Execute the custom graph to upgrade compute hosts:

```
fuel2 graph execute --env <env-ID> --type contrail_upgrade_compute
```

7. Log in to compute nodes and verify output of the `contrail-status` command. You can verify the version of the vRouter package by running `contrail-version` command.

CHAPTER 12

Verify Contrail plugin

To verify your installation after deployment, perform the basic checks described below.

1. Verify that Contrail services are running.

   (a) Login to the Contrail controller node and run `contrail-status` command. All services should be in "active" state:

```
# contrail-status
    == Contrail Control ==
    supervisor-control:           active
    contrail-control              active
    contrail-control-nodemgr      active
    contrail-dns                  active
    contrail-named                active

    == Contrail Analytics ==
    supervisor-analytics:         active
    contrail-analytics-api        active
    contrail-analytics-nodemgr    active
    contrail-collector            active
    contrail-query-engine         active
    contrail-snmp-collector       active
    contrail-topology             active

    == Contrail Config ==
    supervisor-config:            active
    contrail-api:0                active
    contrail-config-nodemgr       active
    contrail-device-manager       active
    contrail-discovery:0          active
    contrail-schema               active
    contrail-svc-monitor          active
    ifmap                         active

    == Contrail Web UI ==
    supervisor-webui:             active
```

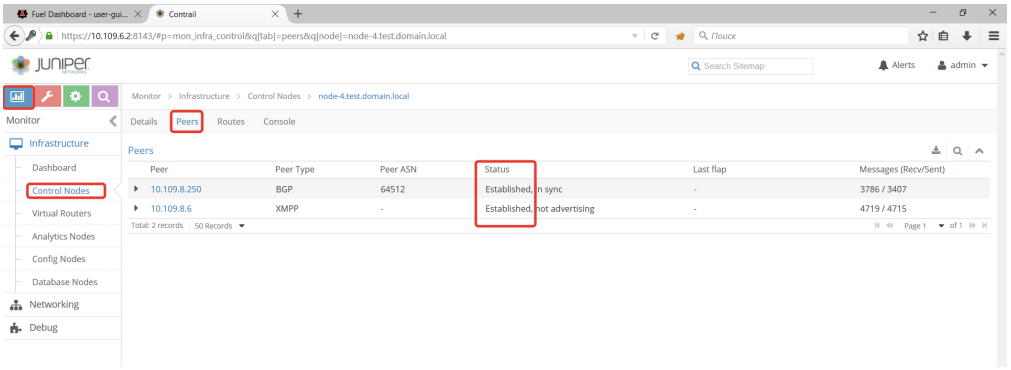
```
contrail-webui                  active
contrail-webui-middleware       active
== Contrail Database ==
supervisor-database:            active
contrail-database               active
contrail-database-nodemgr       active
kafka                           active
```

2. Verify the list of peers and peering status

   (a) Login to Contrail web UI

   (b) Go to *Monitor -> Control nodes*

   (c) Choose any and select a *Peers* tab. You should see your compute nodes (vRouters) and external router in a list of peers with status `Established`

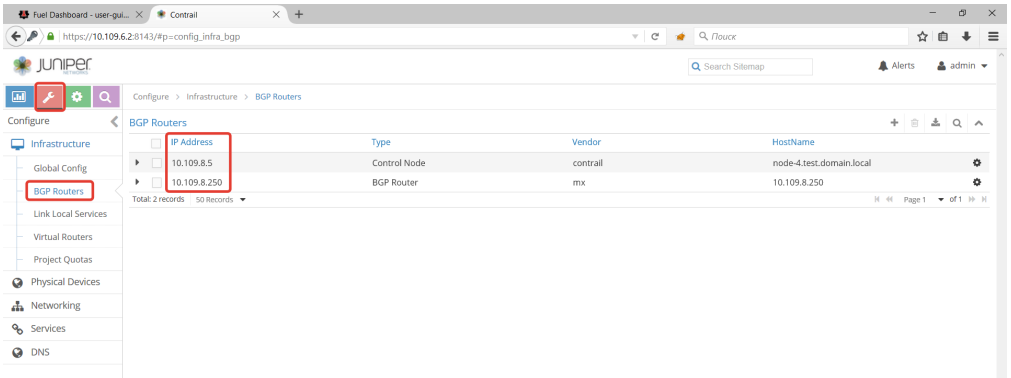


3. Verify that external router has been provisioned correctly:

   (a) Login to Contrail web UI

   (b) Go to *Configure -> Infrastructure -> BGP routers.*

   (c) Verify the IP address of the router



   (a) Use health checks in Fuel web UI, also called OSTF tests.

Run OSTF tests

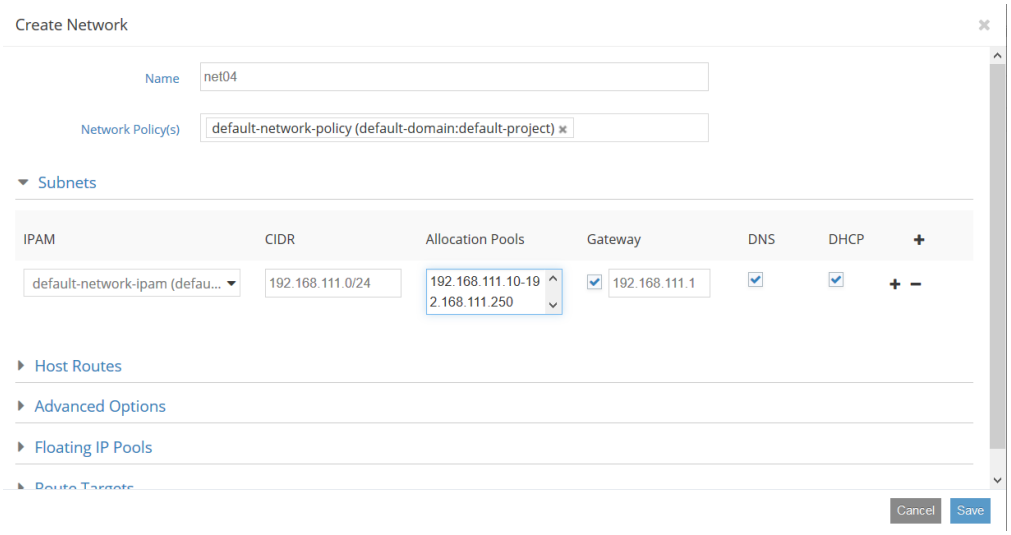
Prerequisites for OSTF

1. OSTF tests require two pre-defined networks created - `admin_internal_net` and `admin_external_net`. The networks are created by Fuel during deployment. This section includes instructions how to create them if they were accidentally deleted. Floating IP addresses from net04_ext should be accessible from Fuel master node.
2. Three tests from `Functional tests` set require floating IP addresses. They should be configured on external router, routable from Fuel master node and populated in the Openstack with Contrail environment.
3. HA tests require at least three Openstack controllers.
4. `Platform services functional tests.` require Ceilometer and MongoDB.

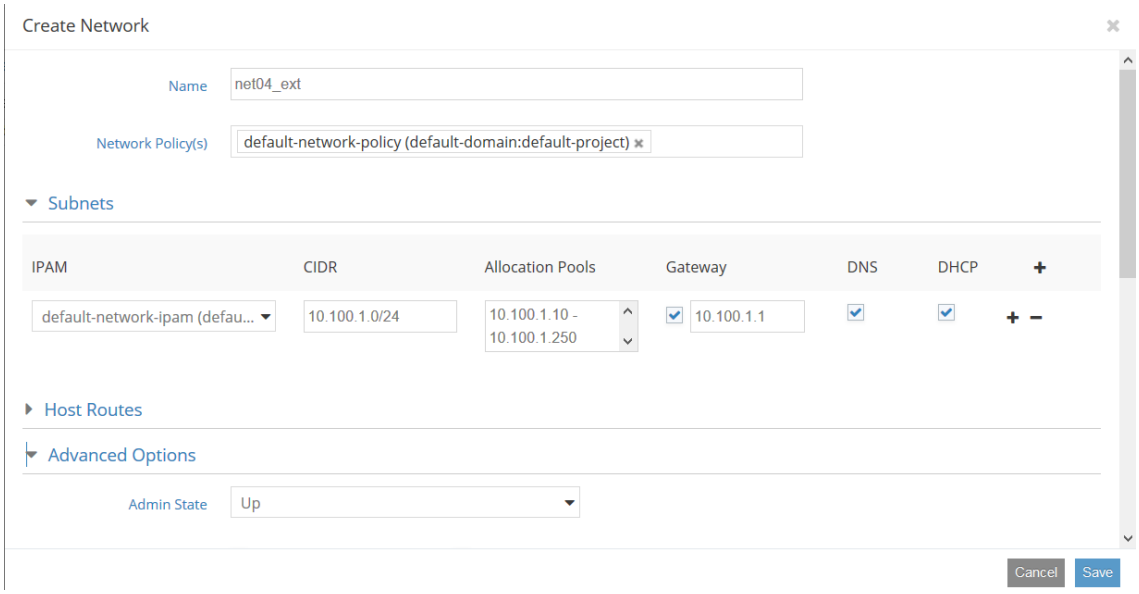
Configure OSTF networks and floating IPs

To configure OSTF networks and floating IPs:

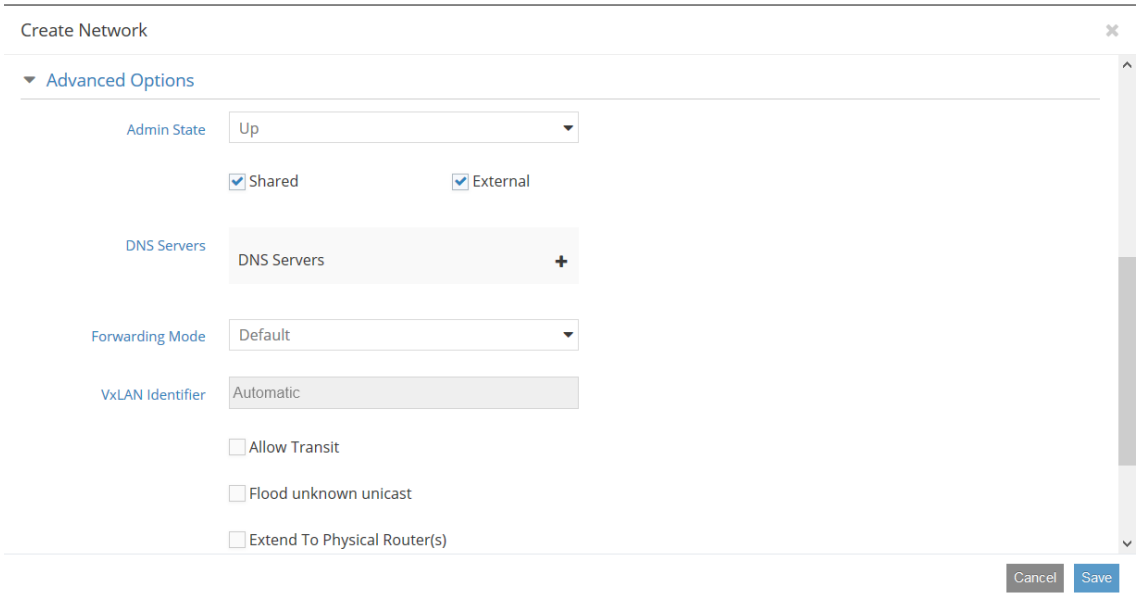
1. Go to Contrail web UI *Configure -> Networking -> Networks*
2. Create network `admin_internal_net`



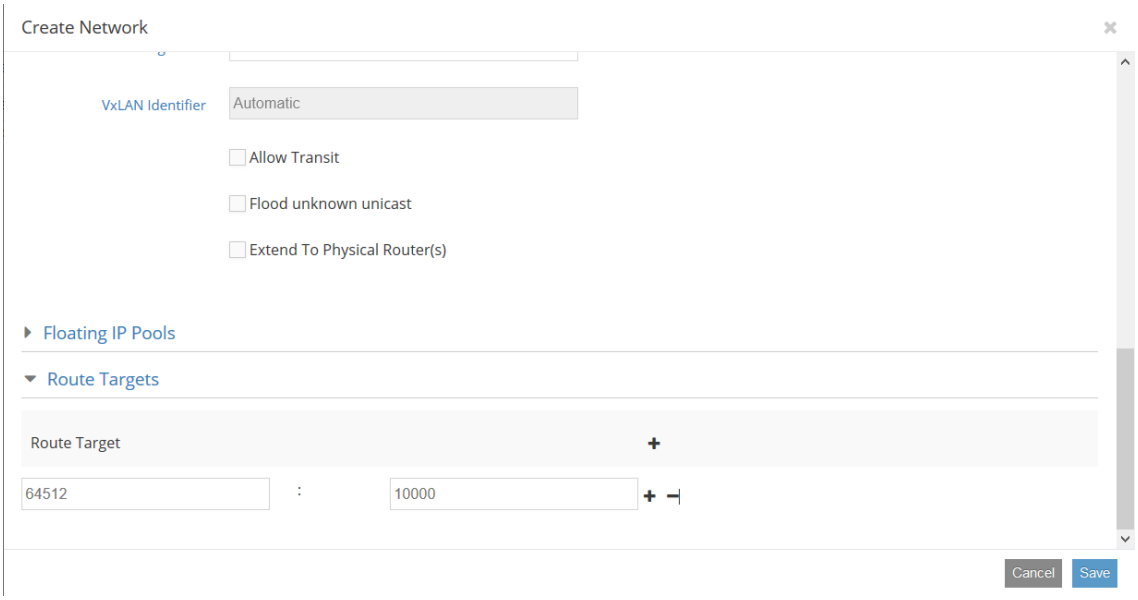
3. Create network `admin_external_net`.



It should be marked as `shared` and `external`

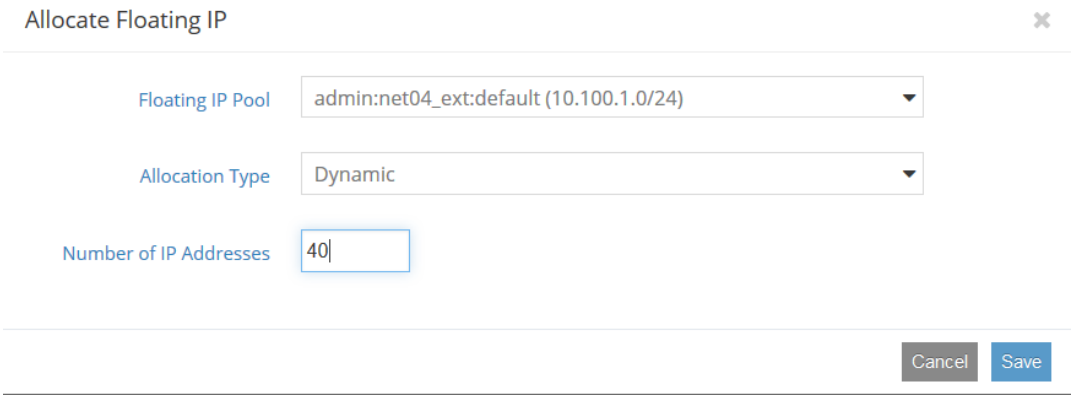


And have same route target as configured in an external router



4. Allocate floating IP addresses from `admin_external_net`

   (a) Go to Contrail WebUI *Configure -> Networking -> Manage Floating IPs*



5. Start OSTF tests.

See also:

Fuel user-guide.

Warning: 'OSTF test 'Check network connectivity from SRIOV instance via floating IP' is expected to fail in environments with Contrail. Contrail doesn't support assigning floating addresses to SRIOV ports, they are treated as pci passthrough devices.

Troubleshooting

To troubleshoot:

1. Verify output of the `contrail-status` command.
2. Verify the logs for corresponding serivice:

- Contrail logs are located in `/var/log/contrail/` directory, and log names match with contrail service name.
- Cassandra logs are located in `/var/log/cassandra/`
- Zookeeper logs are in `/var/log/zookeeper/`

CHAPTER 13

Restore failed Contrail node

This guide describes how to replace the failed Contrail all-in-one node (with all Contrail roles assigned) in a multi-node environment.

If your Contrail node has been crashed, follow the steps to fix the issue:

1. Remove failed node from Cassandra cluster (on working contrail node)
   (a) Obtain Host-ID of the failed Cassandra node:

   ```
   nodetool status
   ```

   (b) Remove the failed node:

   ```
   nodetool removenode <Host-ID>
   ```

2. Deprovision analytics, control, database, and config components of the failed node from contrail db.
   (a) Obtain IP address of Contrail API endpoint (Managment VIP):

   ```
   hiera management_vip
   ```

   Example of system response:

   ```
   10.109.1.3
   ```

   (b) Obtain Neutron service password:

   ```
   hiera neutron_config | grep admin_password
   ```

   Example of system response:

   ```
   "keystone"=>{"admin_password"=>"VerySecurePassword!"},
   ```

   (c) Deprovision `contrail-config`:

```
/opt/contrail/utils/provision_config_node.py \
--api_server_ip <Managment VIP> \
--api_server_port 8082 \
--oper del \
--host_name node-294.domain.tld \
--host_ip 172.21.129.193 \
--admin_user neutron \
--admin_tenant_name services \
--admin_password <Neutron password>
```

(d) Deprovision `contrail-analytics`:

```
/opt/contrail/utils/provision_analytics_node.py \
--api_server_ip <Managment VIP> \
--api_server_port 8082 \
--oper del \
--host_name node-294.domain.tld \
--host_ip 172.21.129.193 \
--admin_user neutron \
--admin_tenant_name services \
--admin_password <Neutron password>
```

(e) Deprovision `contrail-control`:

```
/opt/contrail/utils/provision_control.py \
--api_server_ip <Managment VIP> \
--api_server_port 8082 \
--oper del \
--host_name  node-294.domain.tld \
--host_ip 172.21.129.193 \
--router_asn 64512 \
--admin_user neutron \
--admin_tenant_name services \
--admin_password <Neutron password>
```

(f) Deprovision `contrail-database`:

```
/opt/contrail/utils/provision_database_node.py \
--api_server_ip <Managment VIP> \
--api_server_port 8082 \
--oper del \
--host_name node-294.domain.tld \
--host_ip 172.21.129.193 \
--admin_user neutron \
--admin_tenant_name services \
--admin_password <Neutron password>
```

3. Add a new node with Contrail roles and deploy it with Fuel

CHAPTER 14

Use Contrail

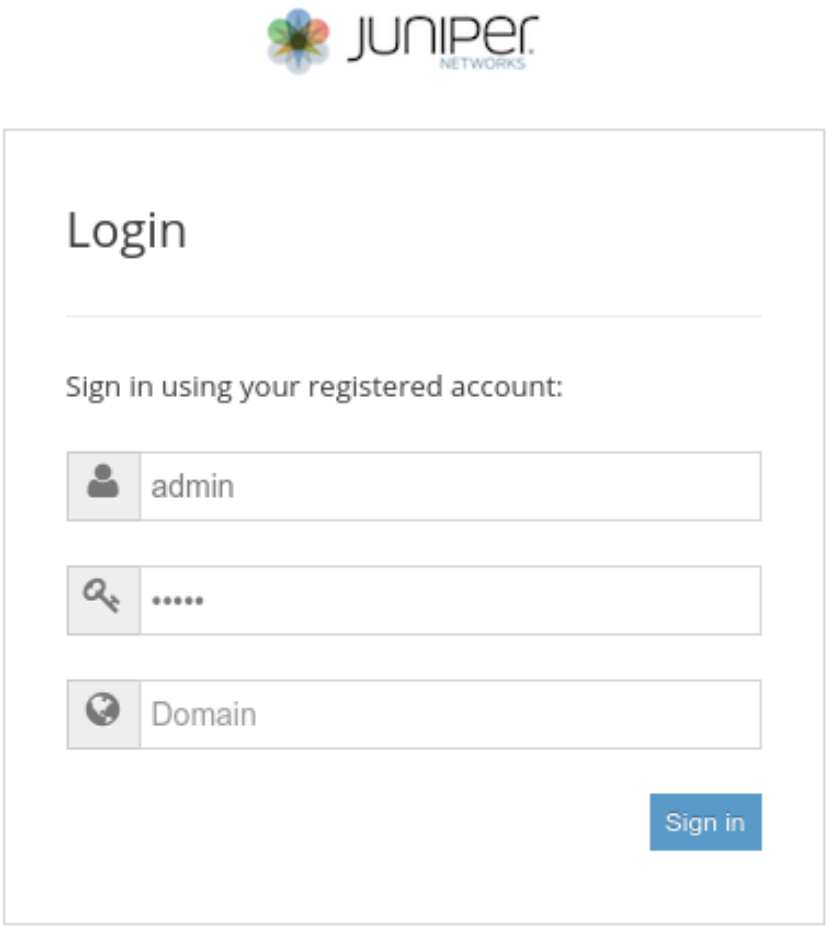
This document describes very basic operations with Contrail UI.

See also:

Juniper documentation.

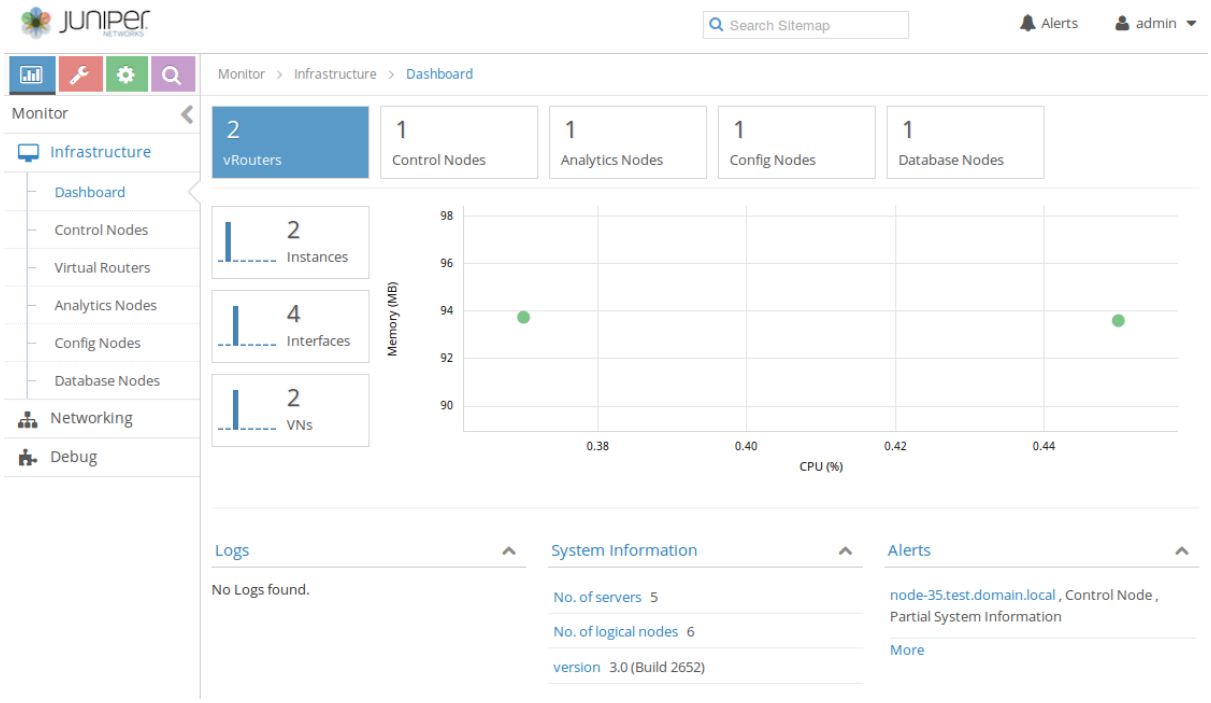
Log into Contrail

To log into Contrail web UI, use the OpenStack admin credentials.



Verify services status

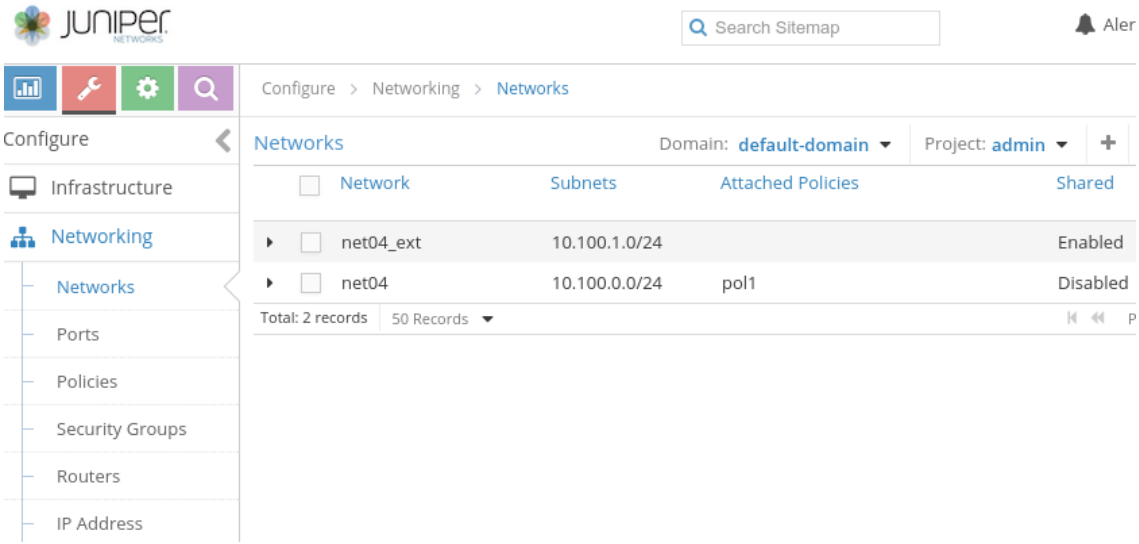
Verify the status of Contrail Control Analytics and Config nodes along with vRouters in *Infrastructure* using *Dashboard* tab of the left-hand *Monitor* menu.



Create the virtual networks

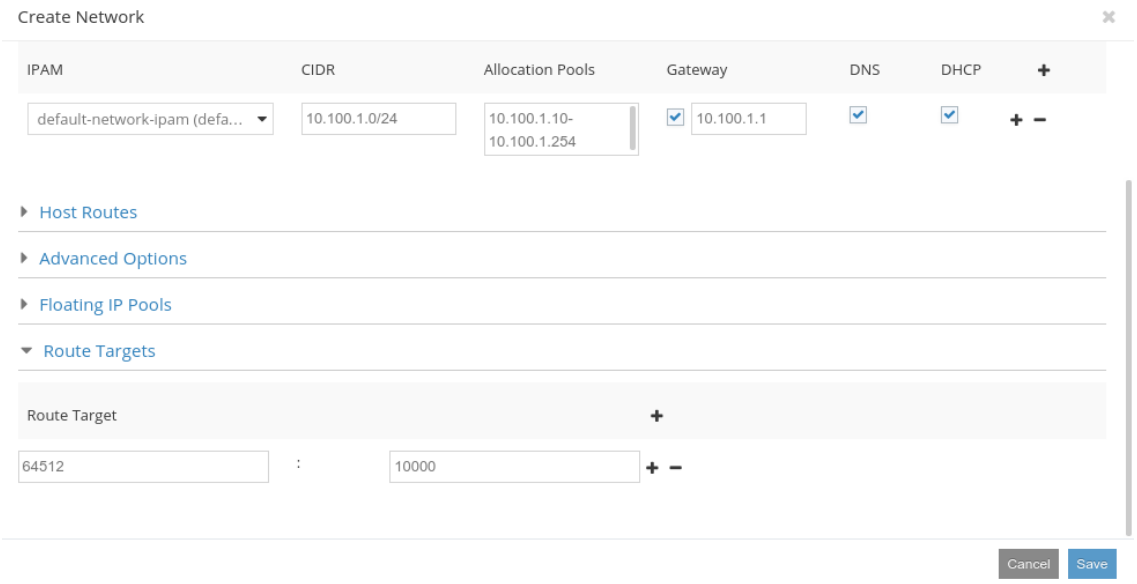
To create the virtual networks:

- Open left-hand *Configure* menu and click *Networking* option. Enter *Networks* tab and use + sign at the right side to create a new virtual network. Enter the network name and add an IP subnet. Gateway address will be added automatically.



- To create an external network, you need to add `Shared` and `External` flags to the created network using the `Advanced Options` sections and provide a proper Routing mark in

Route Targets section to let this network to be announced to the public routing table. The Routing mark is two numbers divided by a semicolon, for example `64512:10000`.



CHAPTER 15

Appendix

1. Contrail overview.
2. Contrail overview at github.com.
3. Contrail major components.
4. Contrail architecture.
5. Contrail quick start guide.