Table of Contents

[JSTACK Setup 3](#_Toc430908720)

[General 3](#_Toc430908721)

[Pre-requisites 3](#_Toc430908722)

[Single Node Setup (all-in-one) 4](#_Toc430908723)

[Node (host) Configuration 4](#_Toc430908724)

[Set MTU Value 5](#_Toc430908725)

[Multi Node Virtualized (Nested) Setup 5](#_Toc430908726)

[Controller Node 6](#_Toc430908727)

[Network Node 6](#_Toc430908728)

[Compute Node 7](#_Toc430908729)

[**Note:** Please refer to the Pre-requisites screenshot and Make sure your compute VM has the following settings enabled: 8](#_Toc430908730)

[Completing the Installation 8](#_Toc430908731)

[Troubleshooting 8](#_Toc430908732)

[Restarting 8](#_Toc430908733)

[DHCP Checksum issue 9](#_Toc430908734)

[NSC Setup 9](#_Toc430908735)

[Installation 9](#_Toc430908736)

[Pre-requisites 9](#_Toc430908737)

[Server 10](#_Toc430908738)

[Agent 10](#_Toc430908739)

[Diagnostic and Troubleshooting 10](#_Toc430908740)

[NSC logs 10](#_Toc430908741)

[NSC Hooks and Inspection Ports state 10](#_Toc430908742)

[OVS Flows (Compute Node) 11](#_Toc430908743)

[Openstack Command Tips 12](#_Toc430908744)

[Managing Floating IP Address Pool 12](#_Toc430908745)

[Listing 12](#_Toc430908746)

[Updating 12](#_Toc430908747)

[Creation 12](#_Toc430908748)

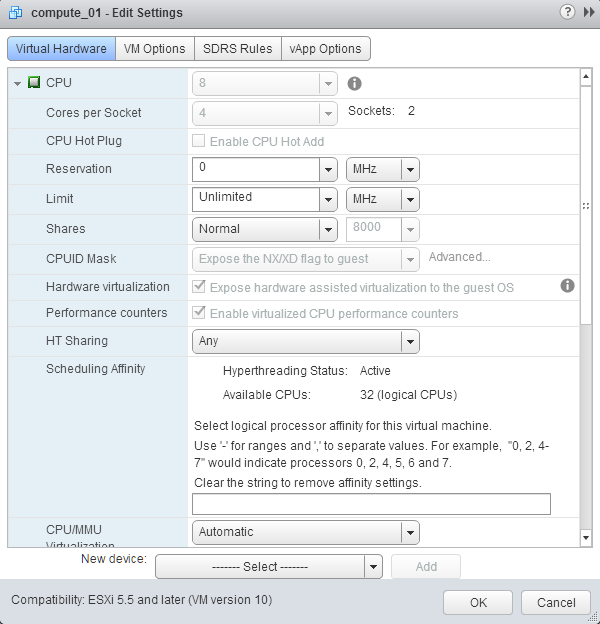
# JSTACK Setup

## General

### Pre-requisites

Before you begin, ensure you have pre-requisites as mention below:

* To allow nested virtualization you need to enable Hardware virtualization on the VM. To do this the VM version needs to be 10 or later and the minimum ESX server version needs to be 5.5 or later.



* There is template Ubuntu image with all the required flags sets with vmware tools and the jstack scripts. The template is located under Controlled\_Specs\vmiDC\Backup\jstack\_vm\_template.
* Compute resources for nested virtualization – recommend at list 16GB RAM and 8 cores on your bare bone physical hypervisor. (Change the configuration of the ovf once deployed.)
* A range of IP’s. You need a set of 5 IP’s. One IP for each node and a contiguous IP range, one for each deployed DAI.

Deploy a VM for each node type you intend to configure (controller, compute, network, or combination of them)

Recommended – once deployed, change root to allow login as root:

* vi /etc/ssh/sshd\_config file. Replace “PermitRootLogin without-password” with “PasswordAuthentication yes”
* sudo passwd root (change the password to admin123 for root)
* sudo chsh -s /bin/bash root
* sudo service ssh restart

The JSTACK Scripts are located in the trunk branch under trunk\tools\jstack. You must copy the latest to each node (Note: You can use winscp to copy the files)

These are a set of 2 scripts which would setup a Juno OpenStack. The scripts need to be run on Ubuntu server 14 or later machines as the root user.

Make sure to run the jstack.sh script as a root user.

The script downloads and installs all the required packages like mysql, neutron, nova, rabbitmq, etc

## Single Node Setup (all-in-one)

### Node (host) Configuration

You can use the template hardware configuration as is. 8 CPU’s with 16GB Memory and a 200GB Harddisk and **two NICs.**

Change the hostname in following files:

/etc/hostname

/etc/hosts

And set the IP address information by modifying the “/etc/network/interfaces” file.

Navigate to /home/vmidc/jstack

The command to install is

./jstack.sh install all

Follow the prompts and answer questions. Note “METERING\_SECRET []” is the ceilometer RabbitMQ default password. It should be **admin123**

After JSTACK installation, make sure compute node is configured to use KVM VM emulation (default is QEMU). Otherwise VMs will run very slow. To do this, modify the following file on compute node and ensure it contains:

vi /etc/nova/nova-compute.conf:

[libvirt]

virt\_type = kvm

### Set MTU Value

After the installation is complete, modify the dnsmasq config file to provide MTU value of 1454 for all the vm’s which will be booted in this environment

vi /etc/neutron/dnsmasq-neutron.conf

Add the following line to the file

dhcp-option-force=26,1454

and restart the neutron-dhcp-agent service

initctl restart neutron-dhcp-agent

**After this please complete the steps on page 8.**

All other options can be seen by running

./jstack.sh

You can restart, get status and reconfigure services using the same script.

To de install you can run

./jstack.sh install deinstall

## Multi Node Virtualized (Nested) Setup

To setup a multi node openstack cluster you would need a range of IP’s. You need a set of 5 IP’s to setup a 2 compute node openstack cluster with external connectivity.

To allow nested virtualization you need to enable Hardware virtualization on the VM. To do this the VM version needs to be 10 or later and the minimum ESX server version needs to be 5.5 or later.

There is template Ubuntu image with all the required flags sets with vmware tools and the jstack scripts.

The template is located under Controlled\_Specs\vmiDC\Backup\jstack\_vm\_template

The following nodes needs to be configured

### Controller Node

#### Configuration

You can use the template hardware configuration as is. 8 CPU’s with 4GB Memory and a 200GB Harddisk and single nic.

Change the hostname and set the IP address information by modifying the “/etc/hostname” file and the “/etc/network/interfaces” file

#### Installation

To install the controller run the following command

./jstack.sh install controller

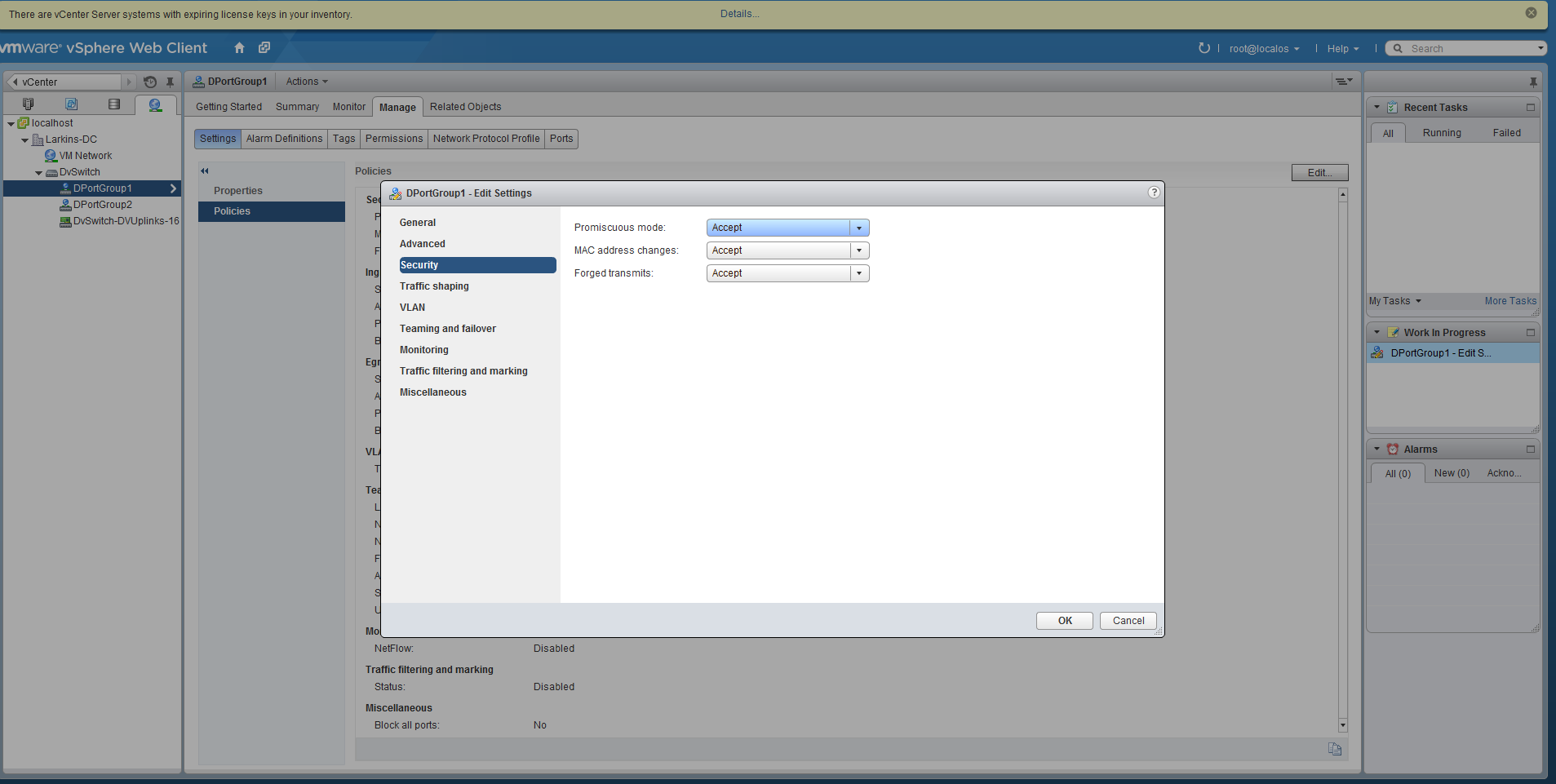
Follow the prompts and provide the required information.

### Network Node

#### Configuration

Use the template as is but add another NIC connected to the VM-network.

Go to the vSpehere WebUI and navigate to the host where the network node is deployed 🡪Manage 🡪 Networking 🡪 Virtual Switches 🡪 Select the vSwitch 🡪 Click on edit for the vSwitch(VM Network) 🡪 Security 🡪 Promiscuous Mode 🡪 Accept



#### Installation

To install the network node run the following command:

./jstack.sh install network

Follow the prompts and provide the required information.

To configure external public access to VMs, run the following command:

./jstack.sh install neutronexternal

Follow the prompts and provide the required information.

#### Set MTU Value

After the installation is complete, modify the dnsmasq config file to provide MTU value of 1454 for all the vm’s which will be booted in this environment

vi /etc/neutron/dnsmasq-neutron.conf

Add the following line to the file

dhcp-option-force=26,1454

and restart the neutron-dhcp-agent service

initctl restart neutron-dhcp-agent

### Compute Node

You can add multiple compute nodes to this cluster by following the same instructions

#### Configuration

Use the template as is but add more memory and CPU as needed.

### **Note:** Please refer to the Pre-requisites screenshot and Make sure your compute VM has the following settings enabled:

* **Hardware Virtualization**
* **Performance counters**

#### Installation

To install the compute node run the following command

./jstack.sh install compute

### Completing the Installation

After all the VM’s are up and running and you verify you can see the compute hosts by going to the controller node and logging in to Horizon.

Go to the controller node and run the following command and follow the prompts

./jstack.sh install rcfiles

./jstack.sh install neutroninternal

./jstack.sh install neutronexternal

This will create the internal and external networks and a router between them.

Any VM’s provisioned on this environment should now have external connectivity.

## Troubleshooting

### Restarting

If you experience issues with VM not being deployed, not receiving notifications, VM not starting, VM errors, try restarting openstack components

#### Controller Node:

./jstack.sh restart controller

**initctl stop nsc-server ; initctl start nsc-server**

#### Compute Node:

./jstack.sh restart compute

**initctl stop neutron-plugin-nsc-agent ; initctl start neutron-plugin-nsc-agent**

#### Network Node:

./jstack.sh restart network

#### All-in-One Node:

./jstack.sh restart controller; ./jstack.sh restart network; ./jstack.sh restart compute

**initctl stop nsc-server ; initctl start nsc-server**

**initctl stop neutron-plugin-nsc-agent ; initctl start neutron-plugin-nsc-agent**

### DHCP Checksum issue

We rely on the internal Openstack DHCP server to provide IP’s for our SVA. There is an issue in openstack where the DHCP service do not work correctly because of a bad checksum calculation issue.

To fix the issue run the following command on your network node 🡪 dhcp name space

ip netns list

ip netns exec qdhcp-e6553557-f1d9-41f3-b6b8-14788ee104c4 iptables -A POSTROUTING -t mangle -p udp --dport 68 -j CHECKSUM --checksum-fill

ip netns exec qdhcp-e6553557-f1d9-41f3-b6b8-14788ee104c4 iptables -t mangle -L

# NSC Setup

## Installation

The NSC Agent files are located in the trunk branch under trunk\tools\isc-agent

Using root user, copy over tar files to root’s home directory.

### Pre-requisites

**apt-get update**

**apt-get -y install python-pip**

**pip install Flask**

### Server

On controller node, copy nsc-server-xxxxxxxx.tgz and run following:

**tar vzxf nsc-server-\*.tgz -C /**

**initctl stop nsc-server ; initctl start nsc-server**

### Agent

On compute node(s), copy nsc-agent-xxxxxxxxx.tgz and run following:

**tar vzxf nsc-agent-\*.tgz -C /**

**initctl stop neutron-plugin-openvswitch-agent**

**initctl stop neutron-plugin-nsc-agent ; initctl start neutron-plugin-nsc-agent**

## Diagnostic and Troubleshooting

### NSC logs

#### On Server

**tail -f /var/log/upstart/nsc-server.log**

#### On Client

**tail -f /var/log/upstart/neutron-plugin-nsc-agent.log**

### NSC Hooks and Inspection Ports state

#### List NSC Installed Inspection Hooks

**curl http://localhost:5555/v1.0/nsc/inspectionhooks**

Example:

root@controller:~/isc-agent# curl http://localhost:5555/v1.0/nsc/inspectionhooks

{

"inspection\_hooks": [

{

"enc\_type": "VLAN",

"id": "c09ba80d-6e8a-447d-a8b3-f407412724db",

"inspected\_port\_id": "3558f685-713f-4ac4-9f6b-07bb71a1e607",

"inspection\_port\_id": "c183d02e-50d5-4a9b-9d66-55221bb9a40b",

"order": "0",

"tag": "1"

}

]

}

#### List NSC on-boarded VNF Inspection Ports

**curl http://localhost:5555/v1.0/nsc/inspectionports**

Example:

root@controller:~/isc-agent# curl http://localhost:5555/v1.0/nsc/inspectionports

{

"inspection\_ports": [

{

"id": "2cfc2db4-d353-425a-8f8f-eef7761bc752",

"inspected\_hooks": [],

"port\_id": "2cfc2db4-d353-425a-8f8f-eef7761bc752"

},

{

"id": "c183d02e-50d5-4a9b-9d66-55221bb9a40b",

"inspected\_hooks": [

{

"enc\_type": "VLAN",

"id": "c09ba80d-6e8a-447d-a8b3-f407412724db",

"inspected\_port\_id": "3558f685-713f-4ac4-9f6b-07bb71a1e607",

"inspection\_port\_id": "c183d02e-50d5-4a9b-9d66-55221bb9a40b",

"order": "0",

"tag": "1"

}

],

"port\_id": "c183d02e-50d5-4a9b-9d66-55221bb9a40b"

},

{

"id": "d2eac66e-b100-4ba6-b758-5b3678d5883e",

"inspected\_hooks": [],

"port\_id": "d2eac66e-b100-4ba6-b758-5b3678d5883e"

},

{

"id": "df32fa17-d2e4-40d0-ae54-0b0b360f947c",

"inspected\_hooks": [],

"port\_id": "df32fa17-d2e4-40d0-ae54-0b0b360f947c"

},

{

"id": "ed10a137-6327-4dd4-992b-2ebd98a93622",

"inspected\_hooks": [],

"port\_id": "ed10a137-6327-4dd4-992b-2ebd98a93622"

}

]

}

### OVS Flows (Compute Node)

To continuously examine flows installed on OVS:

**watch -d "ovs-ofctl dump-flows br-int | grep bade"**

Flows with **cookie=0xbade** are ISC flows.

Example:

root@meni-compute-01:~/isc-agent# ovs-ofctl dump-flows br-int | grep bade

**cookie=0xbade**, duration=1.597s, table=10, n\_packets=0, n\_bytes=0, idle\_age=1, priority=2,in\_port=23,dl\_vlan=1,dl\_vlan\_pcp=2,dl\_dst=01:00:00:00:00:00/01:00:00:00:00:00 actions=mod\_vlan\_vid:0,mod\_vlan\_pcp:0,strip\_vlan,load:0x3->NXM\_NX\_TUN\_ID[],output:10,output:20

**cookie=0xbade**, duration=1.75s, table=120, n\_packets=0, n\_bytes=0, idle\_age=1, priority=5,ip,tun\_id=0x3,dl\_dst=01:00:00:00:00:00/01:00:00:00:00:00 actions=mod\_vlan\_vid:1,mod\_vlan\_pcp:2,output:23,mod\_vlan\_vid:0,mod\_vlan\_pcp:0,strip\_vlan,resubmit(,130)

**cookie=0xbade**, duration=1.679s, table=121, n\_packets=0, n\_bytes=0, idle\_age=1, priority=2,tun\_id=0x3,dl\_dst=01:00:00:00:00:00/01:00:00:00:00:00 actions=output:10,output:20

There should be 3 entries for each on-boarded service function instance and 5 per each redirection (2 ingress, 2 egress, 1 broadcast)

# Openstack Command Tips

## Managing Floating IP Address Pool

### Listing

Listing pools:

**nova floating-ip-pool-list**

Listing range:

**neutron subnet-list --floatingip ext-subnet**

### Updating

**neutron subnet-update ext-subnet --allocation-pools start=10.10.12.200,end=10.10.12.210**

### Creation

**nova-manage floating create --pool=[pool name] --ip\_range=[your new IP]**