JOID Configuration guide

Release draft (1098848)

OPNFV

August 18, 2016

CHAPTER

ONE

JOID INSTALLATION

1.1 Bare Metal Installations:

1.2 Requirements as per Pharos:

1.3 Networking:

Minimum 2 networks

```
1. First for Admin network with gateway to access external network
2. Second for public network to consume by tenants for floating ips
```

NOTE: JOID support multiple isolated networks for data as well as storage. Based on your network options for Openstack.

Minimum 6 physical servers

1. Jump host server:

```
Minimum H/W Spec needed
CPU cores: 16
Memory: 32 GB
Hard Disk: 1(250 GB)
NIC: eth0(Admin, Management), eth1 (external network)
```

2. Node servers (minimum 5):

```
Minimum H/W Spec
CPU cores: 16
Memory: 32 GB
Hard Disk: 1(1 TB) this includes the space for ceph as well
NIC: eth0(Admin, Management), eth1 (external network)
```

NOTE: Above configuration is minimum and for better performance and usage of the Openstack please consider higher spec for each nodes.

Make sure all servers are connected to top of rack switch and configured accordingly. No DHCP server should be up and configured. Only gateway at eth0 and eth1 network should be configure to access the network outside your lab.

1.3.1 Jump node configuration:

1. Install Ubuntu 14.04 LTS server version of OS on the nodes. 2. Install the git and bridge-utils packages on the server and configure minimum two bridges on jump host:

brAdm and brPublic cat /etc/network/interfaces

```
# The loopback network interface
auto lo
iface lo inet loopback
iface eth0 inet manual
auto brAdm
iface brAdm inet static
    address 10.4.1.1
    netmask 255.255.248.0
    network 10.4.0.0
    broadcast 10.4.7.255
    gateway 10.4.0.1
    # dns-* options are implemented by the resolvconf package, if installed
    dns-nameservers 10.4.0.2
    bridge_ports eth0
auto brPublic
iface brPublic inet static
    address 10.2.66.2
    netmask 255.255.255.0
    bridge_ports eth2
```

NOTE: If you choose to use the separate network for management, data and storage then you need to create bridge for each interface. In case of VLAN tags use the appropriate network on jump-host depend upon VLAN ID on the interface.

1.4 Configure JOID for your lab

Get the joid code from gerritt

git clone https://gerrit.opnfv.org/gerrit/p/joid.git

Enable MAAS (labconfig.yaml is must and base for MAAS installation and scenario deployment)

If you have already enabled maas for your environment and installed it then there is no need to enabled it again or install it. If you have patches from previous MAAS enablement then you can apply it here.

NOTE: If MAAS is pre installed without 00-maasdeploy.sh then please do the following and skip rest of the step to enable MAAS.

1. Copy MAAS API key and paste in ~/.juju/environments.yaml at appropriate place.
2. Run command cp ~/.juju/environments.yaml ./joid/ci/
3. cp joid/labconfig/<company name>/<pod number>/labconfig.yaml joid/ci/
4. cd joid/ci
5. python deploy.py

If enabling first time then follow it further. - Create a directory in joid/labconfig/<company name>/<pod number>/ for example

mkdir joid/labconfig/intel/pod7/

- copy labconfig.yaml from pod6 to pod7

cp joid/labconfig/intel/pod5/ joid/labconfig/intel/pod7/*

4 files will get copied: labconfig.yaml

1.5 labconfig.yaml file

1.5.1 Prerequisite:

1. Make sure Jump host node has been configured with bridges on each interface, so that appropriate MAAS and JUJU bootstrap VM can be created. For example if you have three network admin, data and public then I would suggest to give names like brAdm, brData and brPublic. 2. You have information about the node MAC address and power management details (IPMI IP, username, password) of the nodes used for control and compute node.

1.6 modify labconfig.yaml

This file has been used to configure your maas and bootstrap node in a VM. Comments in the file are self explanatory and we expect fill up the information according to match lab infrastructure information. Sample labconfig.yaml can be found at https://gerrit.opnfv.org/gerrit/gitweb?p=joid.git;a=blob;f=labconfigintel/pod6/labconfig.yaml

***lab:** location: intel racks: - rack: pod5

nodes: - name: rack-5-m1

architecture: x86_64 roles: [network,control] nics: - ifname: eth1

spaces: [public] mac: [”xx:xx:xx:xx:xx:xx”]

power: type: ipmi address: xx.xx.xx.xx user: xxxx pass: xxxx

- name: rack-5-m1 architecture: x86_64 roles: [network,control] nics: - ifname: eth1

  spaces: [public] mac: [”xx:xx:xx:xx:xx:xx”]

  power: type: ipmi address: xx.xx.xx.xx user: xxxx pass: xxxx

- name: rack-5-m1 architecture: x86_64 roles: [network,control] nics: - ifname: eth1

  spaces: [public] mac: [”xx:xx:xx:xx:xx:xx”]

  power: type: ipmi address: xx.xx.xx.xx user: xxxx pass: xxxx

  - name: rack-5-m1 architecture: x86_64 roles: [network,control] nics: - ifname: eth1

    spaces: [public] mac: ["xx:xx:xx:xx:xx:xx"]

    power: type: ipmi address: xx.xx.xx.xx user: xxxx pass: xxxx

  - name: rack-5-m1 architecture: x86_64 roles: [network,control] nics: - ifname: eth1

    spaces: [public] mac: ["xx:xx:xx:xx:xx:xx"]

    power: type: ipmi address: xx.xx.xx.xx user: xxxx pass: xxxx

  floating-ip-range: 10.5.15.6,10.5.15.250,10.5.15.254,10.5.15.0/24 ext-port: "eth1" dns: 8.8.8.8

opnfv: release: c distro: trusty type: nonha openstack: liberty sdncontroller: - type: nosdn storage: - type: ceph

  disk: /srv

feature: odl_l2 spaces: - type: public

  bridge: brPublic cidr: 10.5.15.0/24 gateway: 10.5.15.254 vlan:

- type: external bridge: brExt cidr: gateway: ipaddress: 10.2.117.92 vlan:*

NOTE: If you are using VLAN tagged network then make sure you modify the case $1 section under Enable vlan interface with maas appropriately.

***'intelpod7')** maas refresh enableautomodebyname eth2 AUTO "10.4.9.0/24" compute || true enableautomodebyname eth2 AUTO "10.4.9.0/24" control || true ;;*

1.7 Deployment of OPNFV using JOID:

Once you have done the change in above section then run the following commands to do the automatic deployments.

1.7.1 MAAS Install

After integrating the changes as mentioned above run the MAAS install. Suppose you name the integration lab as intelpod7 then run the below commands to start the MAAS deployment.

```
./00-maasdeploy.sh custom ../labconfig/intel/pod7/labconfig.yaml
```

OPNFV Install

```
./deploy.sh -o mitaka -s odl -t ha -l custom -f none -d xenial
```

NOTE: Possible options are as follows:

***choose which sdn controller to use.** [-s <nosdn|odl|opencontrail|onos>] nosdn: openvswitch only and no other SDN. odl: OpenDayLight Lithium version. opencontrail: OpenContrail SDN can be installed with Juno Openstack today. onos: ONOS framework as SDN.

[-t <nonha|ha|tip>] nonha: NO HA mode of Openstack ha: HA mode of openstack. [-o <juno|liberty>] juno: Juno Openstack liberty: Liberty version of openstack. [-l <default|intelpod5>] etc... default: For virtual deployment where installation will be done on KVM created using ./02-maasdeploy.sh intelpod5: Install on bare metal OPNFV pod5 of Intel lab. intelpod6 orangepod2 .. .. <your pod>: if you make changes as per your pod above then please use that. [-f <ipv6|none>] none: no special feature will be enabled. ipv6: ipv6 will be enabled for tenant in openstack.*

1.7.2 Troubleshoot

By default debug is enabled in script and error messages will be printed on ssh terminal where you are running the scripts.

To Access of any control or compute nodes. juju ssh <service name> for example to login into openstack-dashboard container.

juju ssh openstack-dashboard/0 juju ssh nova-compute/0 juju ssh neutron-gateway/0

By default juju will add the Ubuntu user keys for authentication into the deployed server and only ssh access will be available.