

# Kolla / Habanero / Ubuntu 16.04

## Introduction

This document explains how to deploy a Newton OpenStack instance on Power 8.

## Configuration

Start from a Power system running Ubuntu 16.04 with two NICs . On our system, enP1p3s0f0 is the 'public' interface and enP1p3s0f0 the 'management' interface.

```
vi /etc/network/interfaces

auto enP1p3s0f0
iface enP1p3s0f0 inet static
    dns-nameservers 8.8.8.8
    address 172.31.250.2/24
    mtu 1500

auto enP1p9s0
iface enP1p9s0 inet static
    gateway 192.168.154.1
    dns-nameservers 8.8.8.8
    address 192.168.154.2/24
    mtu 1500
```

## Preparation

Note: all the Kolla installation must be done under 'root'. Modify the /etc/ssh/sshd\_config file.

```
ssh ubuntu@192.168.154.2

sudo vi /etc/ssh/sshd_config
# PermitRootLogin prohibit-password
PermitRootLogin yes

# PermitEmptyPasswords no

# PasswordAuthentication no

sudo service ssh restart

sudo cp /home/ubuntu/.ssh/authorized_keys /root/.ssh/

exit
```

```
ssh root@192.168.154.2
```

Update /etc/hosts

```
192.168.154.2 hnode1-11.pub.pic2.ibm.com hnode1-11
```

Add SMT off to /etc/rc.local

```
vi /etc/rc.local  
    ppc64_cpu --smt=off
```

Update

```
apt-get update  
apt-get upgrade -y  
apt-get install -y bridge-utils
```

Setup the 'public' interface as a bridge

```
vi /etc/network/interfaces  
  
# auto enP1p3s0f0  
# iface enP1p3s0f0 inet static  
#     address 172.31.250.2/24  
#     dns-nameservers 8.8.8.8  
#     mtu 1500  
  
auto enP1p3s0f0  
iface enP1p3s0f0 inet manual  
  
auto br1  
iface br1 inet static  
    address 172.31.250.2/24  
    gateway 172.31.255.254  
    dns-nameservers 8.8.8.8  
    bridge_ports enP1p3s0f0  
    bridge_fd 9  
    bridge_hello 2  
    bridge_maxage 12  
    bridge_stp off  
    mtu 1500
```

Remove gateway from enP1p9s0

```
auto enP1p9s0  
iface enP1p9s0 inet static  
    # gateway 192.168.154.1
```

```
dns-nameservers 8.8.8.8
address 192.168.154.2/24
mtu 1500
```

## Reboot

```
reboot
```

## Install pip

```
apt install python-pip -y
pip install --upgrade pip
```

## Install Docker

```
apt-get install docker.io=1.10.3-0ubuntu6 -y
```

## Enable insecure access to the registry

```
vi /etc/default/docker
DOCKER_OPTS="--insecure-registry 192.168.154.2:4000"
```

## Restart Docker

```
service docker stop && service docker start
```

## Update package

```
pip install -U docker-py
pip install -U python-openstackclient
pip install -U python-neutronclient
apt-get install -y python-dev libffi-dev libssl-dev gcc git
```

## Create the drop-in unit directory for docker.service

```
mkdir -p /etc/systemd/system/docker.service.d
```

## Create the drop-in unit file

```
tee /etc/systemd/system/docker.service.d/kolla.conf << 'EOF'
[Service]
MountFlags=shared
EOF
```

## Reload and restart Docker

```
systemctl daemon-reload
systemctl restart docker
```

## Mount

```
mount --make-shared /run
```

## Install NTP

```
apt-get install ntp -y
```

## Install Ansible

```
pip install -U ansible==2.1.2.0
```

## Install Kolla

As Kolla on Power is a work in progress, the current version is a temporary custom version. It's distributed as a self contained images:

```
kolla-11182016.tar
kolla-etc-11182016.tar
kolla-tools-11182016.tar
kolla-usr-11182016.tar
```

The `kolla-11182016.tar` file will expand in your local home directory under `kolla`. It contains the original Kolla code from github.

```
tar -xvf kolla-11182016.tar
pip install kolla/
```

Make sure that Kolla version is '2.0.0.0rc2.dev624'

```
pip list | fgrep kolla
kolla (2.0.0.0rc2.dev624)
```

The `kolla-etc-11182016.tar` will expand in the `etc` local directory. It contains the Kolla deployment files specific to your environment. After configuration, these files need to be moved under the `/etc/kolla` directory.

```
tar -xvf kolla-etc-11182016.tar
mkdir -p /etc/kolla
cp etc/kolla/* /etc/kolla
```

Remove the `kolla-build.conf` file as we build from the master.

```
rm /etc/kolla/kolla-build.conf
```

The `kolla-tools-11182016.tar` will expand in your local directory. It's a set of tools to ease the build, deployment, cleanup tasks. They have to be reconfigured to match your local settings.

```
tar -xvf kolla-tools-11182016.tar
```

*Note: after a system reboot run the `after-reboot` script to setup the environment. For automatic setup you can add the content of this file to your `/etc/rc.local` file.*

*Note: most of the tools don't have the IP addresses properly configured. Before running them, please make sure that the IP addresses match your environment.*

The `kolla-usr-11182016.tar` will expand to the `usr` local directory. This is the modified Kolla code. It need to be moved under `/usr`

```
tar -xvf kolla-usr-11182016.tar
mkdir -p /usr/local/share/kolla
cp -R usr/local/share/kolla/* /usr/local/share/kolla
```

### Create an Ubuntu base image for POWER

```
apt-get install -y debootstrap

curl -o debootstrap.sh
https://raw.githubusercontent.com/docker/docker/master/contrib/m
kimage/debootstrap

chmod 755 debootstrap.sh

./debootstrap.sh ubuntu --components=main,universe xenial

tar -C ubuntu -c . | sudo docker import - ubuntu:16.04

docker tag ubuntu:16.04 ubuntu:ppc64

docker images

docker tag <image id> 192.168.154.2:4000/ubuntu:ppc64

I.e docker tag b7ed97810eff 192.168.154.2:4000/ubuntu:ppc64
```

### Install and start registry

```
apt-get install docker-registry -y

vi /etc/docker/registry/config.yml
http:
  addr: :4000
  headers:
    X-Content-Type-Options: [nosniff]

nohup sudo /usr/bin/docker-registry
/etc/docker/registry/config.yml &
```

This will start the registry service on port 4000. Check that it's opened for request

```
nc -zv <registry_host>:<port>
```

i.e.

```
nc -zv 192.168.154.2 4000
Connection to 192.168.154.2 4000 port [tcp/*] succeeded!
```

Push the Ubuntu ppc64 image in the local registry

```
docker push 192.168.154.2:4000/ubuntu:ppc64
```

Building Container Images (takes some time ~40 minutes)

```
kolla-build --base-image 192.168.154.2:4000/ubuntu --base ubuntu
--base-tag ppc64 --type source --registry 192.168.154.2:4000
--push
```

*Note: the 'gnocchi' package does not build (yet) on Power.*

*Note: from time to time, when the network is not reliable, some packages fail to build. In this case just restart the build process.*

After the build completes you can list the images:

```
docker images
```

## Cinder

The cinder implementation defaults to using LVM storage. The default implementation requires a volume group be set up. This can either be a real physical volume or a loopback mounted file for development.

During development, it may be desirable to use file backed block storage. It is possible to use a file and mount it as a block device via the loopback system.

```
mknod /dev/loop2 b 7 2
dd if=/dev/zero of=/var/lib/cinder_data.img bs=1G count=200
losetup /dev/loop2 /var/lib/cinder_data.img
pvcreate /dev/loop2
vgcreate cinder-volumes /dev/loop2
```

## Deployment

Create a pair of virtual interfaces

```
ip link add type veth
brctl addif br1 veth0
ifconfig veth0 up
ifconfig veth1 up
```

## Deploying Kolla (all-in-one)

All variables for the environment can be specified in the files: “/etc/kolla/globals.yml” and “/etc/kolla/passwords.yml”

### Generate new passwords

```
kolla-genpwd
```

### Start by editing the globals.yml file

```
sudo vi /etc/kolla/globals.yml

config_strategy: "COPY_ALWAYS"
kolla_base_distro: "ubuntu"
kolla_install_type: "source"
enable_haproxy: "no"
kolla_internal_vip_address: "192.168.154.2"
kolla_internal_fqdn: "{{ kolla_internal_vip_address }}"
kolla_external_vip_address: "{{ kolla_internal_vip_address }}"
kolla_external_fqdn: "{{ kolla_external_vip_address }}"
docker_registry: "192.168.154.2:4000"
network_interface: "enPlp9s0"
neutron_external_interface: "veth1"
neutron_plugin_agent: "linuxbridge"
enable_heat: "yes"
enable_neutron: "yes"
enable_cinder: "yes"
# enable_iscsi: "yes"
cinder_iscsi_ip_address: "192.168.154.2"
cinder_volume_group: "cinder-volumes"
cinder_volume_backend_name: "lvm"
```

### Run the deployment

```
kolla-ansible deploy
```

After successful deployment of OpenStack, run the following command can create an openrc file /etc/kolla/admin-openrc.sh on the deploy node.

```
kolla-ansible post-deploy
```

### Update configuration script

```
vi /etc/kolla/init-runonce
```

```
# Test to ensure configure script is run only once
if glance image-list | grep -q ubuntu; then
    echo "This tool should only be run once per
deployment."
    exit
fi

echo Downloading glance image.
IMAGE_URL=http://cloud-images.ubuntu.com/xenial/current/
IMAGE=xenial-server-cloudimg-ppc64el-disk1.img
if ! [ -f "$IMAGE" ]; then
    curl -L -o ./ $IMAGE $IMAGE_URL/$IMAGE
fi
echo Creating glance image.
glance image-create --name ubuntu-16.04 --progress
--disk-format qcow2 --container-format bare --progress
--file ./ $IMAGE

echo Configuring neutron.
neutron net-create public1 --router:external
--provider:physical_network physnet1
--provider:network_type flat
neutron subnet-create --name 1-subnet --disable-dhcp
--allocation-pool start=172.31.250.150,end=172.31.250.199
public1 172.31.0.0/16 --gateway 172.31.255.254
--dns-nameservers list=true 8.8.8.8
neutron net-create demo-net --provider:network_type vxlan
neutron subnet-create demo-net 10.0.0.0/24 --name
demo-subnet --gateway 10.0.0.1 --dns-nameservers list=true
8.8.8.8
neutron router-create demo-router
neutron router-interface-add demo-router demo-subnet
neutron router-gateway-set demo-router public1
```

### Run the configuration script

```
source /etc/kolla/admin-openrc.sh
/etc/kolla/init-runonce
```

### Logon to the Horizon dashboard

From the local system:

```
ssh -L 8080:192.168.154.2:80 root@172.31.250.2
```



## Point a Web Browser at

```
http://localhost:8080/auth/login

domain: default
user: admin
password: <from /etc/kolla/admin-openrc.sh>
```

## Add compute node

Logon to the system that needs to be added to the infrastructure as compute node (IP: 192.168.154.4).

```
vi /etc/rc.local
    ppc64_cpu -smt=off
```

Edit /etc/sysctl.conf

```
vi /etc/sysctl.conf
    net.ipv4.conf.all.rp_filter=0
    net.ipv4.conf.default.rp_filter=0

sysctl -p
```

Add Newton ppa (or install a different ppa if Mitaka or Kilo)

```
add-apt-repository ppa:ubuntu-cloud-archive/newton-staging
apt-get update
```

Install nova and neutron

```
apt-get install nova-compute
apt-get install neutron-linuxbridge-agent
```

Stop the services

```
service nova-compute stop
service neutron-linuxbridge-agent stop
```

Configure the services

Download some configuration files from the Kolla controller system

```
cd ~
scp root@192.168.154.2:/etc/kolla//nova-compute/nova.conf .
scp
root@192.168.154.2:/etc/kolla//neutron-linuxbridge-agent/neutron
.conf .
```

```
scp
root@192.168.154.2:/etc/kolla//neutron-linuxbridge-agent/ml2_conf.ini .
```

## Update the configuration files

```
vi ml2_conf.ini

[ml2]
type_drivers = flat,vlan,vxlan
tenant_network_types = vxlan
mechanism_drivers = linuxbridge,l2population

[ml2_type_vlan]
network_vlan_ranges =

[ml2_type_flat]
flat_networks = physnet1

[ml2_type_vxlan]
vni_ranges = 1:1000
vxlan_group = 239.1.1.1

[securitygroup]
firewall_driver =
neutron.agent.linux.iptables_firewall.IptablesFirewallDriver

[linux_bridge]
physical_interface_mappings = physnet1:veth1

[vxlan]
l2_population = true
local_ip = 192.168.154.4

vi neutron.conf

[DEFAULT]
debug = False
log_dir = /var/log/kolla/neutron
use_stderr = False
# bind_host = 192.168.154.2
bind_host = 0.0.0.0
bind_port = 9696
api_paste_config = /usr/share/neutron/api-paste.ini
endpoint_type = internalURL
metadata_proxy_socket =
/var/lib/neutron/kolla/metadata_proxy
interface_driver =
neutron.agent.linux.interface.BridgeInterfaceDriver
```

```
allow_overlapping_ips = true
core_plugin = ml2
service_plugins = router

[nova]
auth_url = http://192.168.154.2:35357
auth_type = password
project_domain_id = default
user_domain_id = default
region_name = RegionOne
project_name = service
username = nova
password = KXZU0G0p68HukxqHq8UI86U7kxzJuSq8HGvbixK
endpoint_type = internal

[oslo_concurrency]
lock_path = /var/lib/neutron/tmp

[oslo_messaging_rabbit]
rabbit_userid = openstack
rabbit_password = bstpqHG52O8sfYl21JYWfz7ReA8Jg6KXmr9bggcA
rabbit_ha_queues = true
rabbit_hosts = 192.168.154.2:5672

[agent]
root_helper = sudo neutron-rootwrap
/etc/neutron/rootwrap.conf

[database]
connection =
mysql+pymysql://neutron:Xlo7xeMLcsmubQma277RfOeh8ULRJYomn9w
698oj@192.168.154.2:3306/neutron
max_retries = -1

[keystone_auth_token]
auth_uri = http://192.168.154.2:5000
auth_url = http://192.168.154.2:35357
auth_type = password
project_domain_id = default
user_domain_id = default
project_name = service
username = neutron
password = TkK0fbFRm4IFFMulNt93r224o1XhqPej1CE9f5HY
memcache_security_strategy = ENCRYPT
memcache_secret_key =
mojgGPLte3GcJ5dto67zXhrEWlVOpreBRdf1Ev4l
memcached_servers = 192.168.154.2:11211

[oslo_messaging_notifications]
```

```
driver = noop

vi nova.conf
[DEFAULT]
debug = False
log_dir = /var/log/kolla/nova
use_forwarded_for = true
api_paste_config = /etc/nova/api-paste.ini
state_path = /var/lib/nova
osapi_compute_listen = 0.0.0.0
osapi_compute_listen_port = 8774
osapi_compute_workers = 4
metadata_listen = 0.0.0.0
metadata_listen_port = 8775
metadata_workers = 4
ec2_listen = 0.0.0.0
ec2_listen_port = 8773
use_neutron = True
firewall_driver = nova.virt.firewall.NoopFirewallDriver
scheduler_max_attempts = 10
linuxnet_interface_driver =
nova.network.linux_net.NeutronLinuxBridgeInterfaceDriver
allow_resize_to_same_host = true
compute_driver = libvirt.LibvirtDriver
my_ip = 192.168.154.4

[vnc]
novncproxy_host = 0.0.0.0
enabled = True
novncproxy_port = 6080
vncserver_listen = 0.0.0.0
vncserver_proxyclient_address = 192.168.154.4
novncproxy_base_url =
http://192.168.154.2:6080/vnc_auto.html

[oslo_messaging_rabbit]
rabbit_userid = openstack
rabbit_password = bstpqHG52O8sfYl21JYWfz7ReA8Jg6KXmr9bggcA
rabbit_ha_queues = true
rabbit_hosts = 192.168.154.2:5672

[oslo_concurrency]
lock_path = /var/lib/nova/tmp

[glance]
api_servers = http://192.168.154.2:9292
num_retries = 1

[cinder]
```

```
catalog_info = volume:cinder:internalURL
[neutron]
url = http://192.168.154.2:9696
auth_strategy = keystone
metadata_proxy_shared_secret =
pBsCzjrZ8XGTEWLurBm8ywsEPw8T130RusiLOR2q
service_metadata_proxy = true
auth_url = http://192.168.154.2:35357
auth_type = password
project_domain_name = default
user_domain_id = default
project_name = service
username = neutron
password = TkK0fbFRm4IFFMulNt93r224o1XhqPej1CE9f5HY

[database]
connection =
mysql+pymysql://nova:5GnBhO3QoXQP3Ly1mL30UH27x7GBhRFJ2akyeq
Km@192.168.154.2:3306/nova
max_pool_size = 50
max_overflow = 1000
max_retries = -1

[api_database]
connection =
mysql+pymysql://nova_api:4sfRFI8uOH094YDmhVj8Wgoo5sWAIkFRqY
hrsgta@192.168.154.2:3306/nova_api
max_retries = -1

[cache]
backend = oslo_cache.memcache_pool
enabled = True
memcache_servers = 192.168.154.2:11211

[keystone_auth token]
auth_uri = http://192.168.154.2:5000
auth_url = http://192.168.154.2:35357
auth_type = password
project_domain_id = default
user_domain_id = default
project_name = service
username = nova
password = KXZU00G0p68HukxqHq8UI86U7kxzJuSq8HGvbixK
memcache_security_strategy = ENCRYPT
memcache_secret_key =
mojgGPLte3GcJ5dto67zXhrEWlVOpreBRdf1Ev4l
memcached_servers = 192.168.154.2:11211

# [libvirt]
```

```

# connection_uri = "qemu+tcp://192.168.154.4/system"

[upgrade_levels]
compute = auto

[oslo_messaging_notifications]
driver = noop

[conductor]
workers = 4

vi nova-compute.conf

[DEFAULT]
[libvirt]
virt_type=qemu

```

### Load the configuration files

```

cat ml2_conf.ini >>
/etc/neutron/plugins/ml2/linuxbridge_agent.ini

cp /etc/neutron/neutron.conf /etc/neutron/neutron.conf.orig

cp neutron.conf /etc/neutron/neutron.conf

cp /etc/nova/nova.conf /etc/nova/nova.conf.orig

cp nova.conf /etc/nova/nova.conf

cp nova-compute.conf /etc/nova/nova-compute.conf

```

### Start the service manually

```

/usr/bin/python /usr/bin/nova-compute
--config-file=/etc/nova/nova.conf
--config-file=/etc/nova/nova-compute.conf
-log-file=/var/log/nova/nova-compute.log

/usr/bin/python /usr/bin/neutron-linuxbridge-agent
--config-file=/etc/neutron/neutron.conf
--config-file=/etc/neutron/plugins/ml2/linuxbridge_agent.ini
--log-file=/var/log/neutron/neutron-linuxbridge-agent.log

```

### Kill the services

#### Restart the services

```

service nova-compute start

service neutron-linuxbridge-agent start

```

## Troubleshooting

If nova-compute logs (docker logs nova-compute) show a 'permission denied error' on KVM:

```
chmod 777 /dev/kvm
```

## Test the registry

```
wget http://192.168.154.2:4000/v2/_catalog
wget http://192.168.154.2:4000/v2/ubuntu/tags/list

docker pull 192.168.154.2:4000/ubuntu:ppc64
ppc64: Pulling from ubuntu
Digest:
sha256:2af24e0b4f901bfe26ca059c28099b939287c7c73bbe8c26b1dfc06b3
472f2ba
Status: Image is up to date for 192.168.154.2:4000/ubuntu:ppc64

docker run -t -i 192.168.154.2:4000/ubuntu:ppc64 bash
root@e142a20f5597:/#
```

## Test Cinder

```
openstack volume create --size 1 frb-test-vol
```

Field	Value
attachments	[]
availability_zone	nova
bootable	false
consistencygroup_id	None
created_at	2016-11-18T14:50:54.735141
description	None
encrypted	False
id	2a442bcc-533c-4909-9912-283acc040c5b
migration_status	None
multiattach	False
name	frb-test-vol
properties	
replication_status	disabled
size	1
snapshot_id	None
source_volid	None
status	creating
type	None
updated_at	None
user_id	24bf3349e7bb4cefbbd6c8a407003f76

## openstack volume list

ID	Display Name	Status	Size	Attached to
----	--------------	--------	------	-------------

2a442bcc-533c-4909-9912-283acc040c5b	frb-test-vol	available	1
--------------------------------------	--------------	-----------	---