OPNFV Installation instructions (Apex)

Release draft (768da1a)

OPNFV

August 21, 2016

CONTENTS

Contents:

CHAPTER

ONE

ABSTRACT

This document describes how to install the Colorado release of OPNFV when using Apex as a deployment tool covering it's limitations, dependencies and required system resources.

CHAPTER

TWO

LICENSE

CHAPTER

THREE

INTRODUCTION

This document describes the steps to install an OPNFV Colorado reference platform, as defined by the Genesis Project using the Apex installer.

The audience is assumed to have a good background in networking and Linux administration.

CHAPTER

FOUR

PREFACE

Apex uses Triple-O from the RDO Project OpenStack distribution as a provisioning tool. The Triple-O image based life cycle installation tool provisions an OPNFV Target System (3 controllers, 2 or more compute nodes) with OPNFV specific configuration provided by the Apex deployment tool chain.

The Apex deployment artifacts contain the necessary tools to deploy and configure an OPNFV target system using the Apex deployment toolchain. These artifacts offer the choice of using the Apex bootable ISO (`opnfv-apex-colorado.iso`) to both install CentOS 7 and the necessary materials to deploy or the Apex RPMs (`opnfv-apex*.rpm`), and their associated dependencies, which expects installation to a CentOS 7 libvirt enabled host. The RPM contains a collection of configuration files, prebuilt disk images, and the automatic deployment script (`opnfv-deploy`).

An OPNFV install requires a "Jumphost" in order to operate. The bootable ISO will allow you to install a customized CentOS 7 release to the Jumphost, which includes the required packages needed to run `opnfv-deploy`. If you already have a Jumphost with CentOS 7 installed, you may choose to skip the ISO step and simply install the (`opnfv-apex*.rpm`) RPMs. The RPMs are the same RPMs included in the ISO and include all the necessary disk images and configuration files to execute an OPNFV deployment. Either method will prepare a host to the same ready state for OPNFV deployment.

`opnfv-deploy` instantiates a Triple-O Undercloud VM server using libvirt as its provider. This VM is then configured and used to provision the OPNFV target deployment (3 controllers, n compute nodes). These nodes can be either virtual or bare metal. This guide contains instructions for installing either method.

CHAPTER

FIVE

TRIPLE-O DEPLOYMENT ARCHITECTURE

Apex is based on the OpenStack Triple-O project as distributed by the RDO Project. It is important to understand the basics of a Triple-O deployment to help make decisions that will assist in successfully deploying OPNFV.

Triple-O stands for OpenStack On OpenStack. This means that OpenStack will be used to install OpenStack. The target OPNFV deployment is an OpenStack cloud with NFV features built-in that will be deployed by a smaller all-in-one deployment of OpenStack. In this deployment methodology there are two OpenStack installations. They are referred to as the undercloud and the overcloud. The undercloud is used to deploy the overcloud.

The undercloud is the all-in-one installation of OpenStack that includes baremetal provisioning capability. The undercloud will be deployed as a virtual machine on a jumphost. This VM is pre-built and distributed as part of the Apex RPM.

The overcloud is OPNFV. Configuration will be passed into undercloud and the undercloud will use OpenStack's orchestration component, named Heat, to execute a deployment that will provision the target OPNFV nodes.

CHAPTER

SIX

OPNFV SCENARIO ARCHITECTURE

OPNFV distinguishes different types of SDN controllers, deployment options, and features into “scenarios”. These scenarios are universal across all OPNFV installers, although some may or may not be supported by each installer.

The standard naming convention for a scenario is: <VIM platform>-<SDN type>-<feature>-<ha/noha>

The only supported VIM type is “OS” (OpenStack), while SDN types can be any supported SDN controller. “feature” includes things like ovs_dpdk, sfc, etc. “ha” or “noha” determines if the deployment will be highly available. If “ha” is used at least 3 control nodes are required.

CHAPTER

SEVEN

OPNFV SCENARIOS IN APEX

Apex provides pre-built scenario files in /etc/opnfv-apex which a user can select from to deploy the desired scenario. Simply pass the desired file to the installer as a (-d) deploy setting. Read further in the Apex documentation to learn more about invoking the deploy command. Below is quick reference matrix for OPNFV scenarios supported in Apex. Please refer to the respective OPNFV Docs documentation for each scenario in order to see a full scenario description. The following scenarios correspond to a supported <Scenario>.yaml deploy settings file:

Scenario	Owner	Known Issues
os-nosdn-nofeature-ha	Apex	
os-nosdn-nofeature-noha	Apex	
os-nosdn-ovs-noha	OVS for NFV	
os-nosdn-fdio-noha	FDS	
os-odl_l2-nofeature-ha	Apex	
os-odl_l3-nofeature-ha	Apex	APEX-112
os-odl_l2-sfc-noha	SFC	
os-odl_l2-bgpvpn-noha	SDNVPN	
os-odl_l2-fdio-noha	FDS	
os-onos-nofeature-ha	ONOSFW	
os-onos-sfc-ha	ONOSFW	

CHAPTER

EIGHT

SETUP REQUIREMENTS

8.1 Jumphost Requirements

The Jumphost requirements are outlined below:

1. CentOS 7 (from ISO or self-installed).
2. Root access.
3. libvirt virtualization support.
4. minimum 1 networks and maximum 5 networks, multiple NIC and/or VLAN combinations are supported. This is virtualized for a VM deployment.
5. The Colorado Apex RPMs and their dependencies.
6. 16 GB of RAM for a bare metal deployment, 64 GB of RAM for a VM deployment.

8.2 Network Requirements

Network requirements include:

1. No DHCP or TFTP server running on networks used by OPNFV.
2. 1-5 separate networks with connectivity between Jumphost and nodes.
    - Control Plane (Provisioning)
    - Private Tenant-Networking Network*
    - External Network
    - Storage Network*
    - Internal API Network* (required for IPv6 **)
3. Lights out OOB network access from Jumphost with IPMI node enabled (bare metal deployment only).
4. External network is a routable network from outside the cloud, deployment. The External network is where public internet access would reside if available.

* ***These networks can be combined with each other or all combined on the** Control Plane network.*

* ***Non-External networks will be consolidated to the Control Plane network** if not specifically configured.*

** ***Internal API network, by default, is collapsed with provisioning in IPv4** deployments, this is not possible with the current lack of PXE boot support and therefore the API network is required to be its own network in an IPv6 deployment.*

8.3 Bare Metal Node Requirements

Bare metal nodes require:

1. IPMI enabled on OOB interface for power control.
2. BIOS boot priority should be PXE first then local hard disk.
3. BIOS PXE interface should include Control Plane network mentioned above.

8.4 Execution Requirements (Bare Metal Only)

In order to execute a deployment, one must gather the following information:

1. IPMI IP addresses for the nodes.
2. IPMI login information for the nodes (user/pass).
3. MAC address of Control Plane / Provisioning interfaces of the overcloud nodes.

CHAPTER

NINE

INSTALLATION HIGH-LEVEL OVERVIEW - BARE METAL DEPLOYMENT

The setup presumes that you have 6 or more bare metal servers already setup with network connectivity on at least 1 or more network interfaces for all servers via a TOR switch or other network implementation.

The physical TOR switches are **not** automatically configured from the OPNFV reference platform. All the networks involved in the OPNFV infrastructure as well as the provider networks and the private tenant VLANs needs to be manually configured.

The Jumphost can be installed using the bootable ISO or by using the (`opnfv-apex*.rpm`) RPMs and their dependencies. The Jumphost should then be configured with an IP gateway on its admin or public interface and configured with a working DNS server. The Jumphost should also have routable access to the lights out network for the overcloud nodes.

`opnfv-deploy` is then executed in order to deploy the undercloud VM and to provision the overcloud nodes. `opnfv-deploy` uses three configuration files in order to know how to install and provision the OPNFV target system. The information gathered under section Execution Requirements (Bare Metal Only) is put into the YAML file `/etc/opnfv-apex/inventory.yaml` configuration file. Deployment options are put into the YAML file `/etc/opnfv-apex/deploy_settings.yaml`. Alternatively there are pre-baked deploy_settings files available in `/etc/opnfv-apex/`. These files are named with the naming convention os-sdn_controller-enabled_feature-[no]ha.yaml. These files can be used in place of the `/etc/opnfv-apex/deploy_settings.yaml` file if one suites your deployment needs. Networking definitions gathered under section Network Requirements are put into the YAML file `/etc/opnfv-apex/network_settings.yaml`. `opnfv-deploy` will boot the undercloud VM and load the target deployment configuration into the provisioning toolchain. This information includes MAC address, IPMI, Networking Environment and OPNFV deployment options.

Once configuration is loaded and the undercloud is configured it will then reboot the overcloud nodes via IPMI. The nodes should already be set to PXE boot first off the admin interface. The nodes will first PXE off of the undercloud PXE server and go through a discovery/introspection process.

Introspection boots off of custom introspection PXE images. These images are designed to look at the properties of the hardware that is being booted and report the properties of it back to the undercloud node.

After introspection the undercloud will execute a Heat Stack Deployment to continue node provisioning and configuration. The nodes will reboot and PXE from the undercloud PXE server again to provision each node using Glance disk images provided by the undercloud. These disk images include all the necessary packages and configuration for an OPNFV deployment to execute. Once the disk images have been written to node's disks the nodes will boot locally and execute cloud-init which will execute the final node configuration. This configuration is largly completed by executing a puppet apply on each node.

CHAPTER

TEN

INSTALLATION HIGH-LEVEL OVERVIEW - VM DEPLOYMENT

The VM nodes deployment operates almost the same way as the bare metal deployment with a few differences mainly related to power management. `opnfv-deploy` still deploys an undercloud VM. In addition to the undercloud VM a collection of VMs (3 control nodes + 2 compute for an HA deployment or 1 control node and 1 or more compute nodes for a Non-HA Deployment) will be defined for the target OPNFV deployment. The part of the toolchain that executes IPMI power instructions calls into libvirt instead of the IPMI interfaces on baremetal servers to operate the power managment. These VMs are then provisioned with the same disk images and configuration that baremetal would be.

To Triple-O these nodes look like they have just built and registered the same way as bare metal nodes, the main difference is the use of a libvirt driver for the power management.

CHAPTER

ELEVEN

INSTALLATION GUIDE - BARE METAL DEPLOYMENT

This section goes step-by-step on how to correctly install and provision the OPNFV target system to bare metal nodes.

11.1 Install Bare Metal Jumphost

1a. If your Jumphost does not have CentOS 7 already on it, or you would like to do a fresh install, then download the Apex bootable ISO from the OPNFV artifacts site <http://artifacts.opnfv.org/apex.html>. There have been isolated reports of problems with the ISO having trouble completing installation successfully. In the unexpected event the ISO does not work please workaround this by downloading the CentOS 7 DVD and performing a "Virtualization Host" install. If you perform a "Minimal Install" or install type other than "Virtualization Host" simply run `sudo yum groupinstall "Virtualization Host" chkconfig libvirtd on && reboot` to install virtualzation support and enable libvirt on boot. If you use the CentOS 7 DVD proceed to step 1b once the CentOS 7 with "Virtualzation Host" support is completed.

1b. If your Jump host already has CentOS 7 with libvirt running on it then install the install the RDO Release RPM:

```
sudo yum install -y https://www.rdoproject.org/repos/rdo-release.rpm
```

The RDO Project release repository is needed to install OpenVSwitch, which is a dependency of opnfv-apex. If you do not have external connectivity to use this repository you need to download the OpenVSwitch RPM from the RDO Project repositories and install it with the opnfv-apex RPM.

2a. Boot the ISO off of a USB or other installation media and walk through installing OPNFV CentOS 7. The ISO comes prepared to be written directly to a USB drive with dd as such:

```
dd if=opnfv-apex.iso of=/dev/sdX bs=4M
```

Replace /dev/sdX with the device assigned to your usb drive. Then select the USB device as the boot media on your Jumphost

2b. If your Jump host already has CentOS 7 with libvirt running on it then install the opnfv-apex RPMs from the OPNFV artifacts site <http://artifacts.opnfv.org/apex.html>. The following RPMS are available for installation:

- opnfv-apex - OpenDaylight L2 / L3 and ONOS support *
- opnfv-apex-onos - ONOS support *
- opnfv-apex-opendaylight-sfc - OpenDaylight SFC support *
- opnfv-apex-undercloud - (reqed) Undercloud Image
- opnfv-apex-common - (reqed) Supporting config files and scripts
- python34-markupsafe - (reqed) Dependency of opnfv-apex-common **

- python3-jinja2 - (reqed) Dependency of opnfv-apex-common **

* One or more of these RPMs is required Only one of opnfv-apex, opnfv-apex-onos and opnfv-apex-opendaylight-sfc is required. It is safe to leave the unneeded SDN controller's RPMs uninstalled if you do not intend to use them.

** These RPMs are not yet distributed by CentOS or EPEL. Apex has built these for distribution with Apex while CentOS and EPEL do not distribute them. Once they are carried in an upstream channel Apex will no longer carry them and they will not need special handling for installation.

To install these RPMs download them to the local disk on your CentOS 7 install and pass the file names directly to yum: `sudo yum install python34-markupsafe-<version>.rpm python3-jinja2-<version>.rpm sudo yum install opnfv-apex-<version>.rpm opnfv-apex-undercloud-<version>.rpm opnfv-apex-common-<version>.rpm`

3. After the operating system and the opnfv-apex RPMs are installed, login to your Jumphost as root.
4. Configure IP addresses on the interfaces that you have selected as your networks.
5. Configure the IP gateway to the Internet either, preferably on the public interface.
6. Configure your `/etc/resolv.conf` to point to a DNS server (8.8.8.8 is provided by Google).

11.2 Creating a Node Inventory File

IPMI configuration information gathered in section Execution Requirements (Bare Metal Only) needs to be added to the `inventory.yaml` file.

1. Copy `/usr/share/doc/opnfv/inventory.yaml.example` as your inventory file template to `/etc/opnfv-apex/inventory.yaml`.
2. The nodes dictionary contains a definition block for each baremetal host that will be deployed. 1 or more compute nodes and 3 controller nodes are required. (The example file contains blocks for each of these already). It is optional at this point to add more compute nodes into the node list.
3. Edit the following values for each node:
   - `mac_address`: MAC of the interface that will PXE boot from undercloud
   - `ipmi_ip`: IPMI IP Address
   - `ipmi_user`: IPMI username
   - `ipmi_password`: IPMI password
   - `pm_type`: Power Management driver to use for the node
   - `cpus`: (Introspected*) CPU cores available
   - `memory`: (Introspected*) Memory available in Mib
   - `disk`: (Introspected*) Disk space available in Gb
   - `arch`: (Introspected*) System architecture
   - `capabilities`: (Opt**) Node role (profile:control or profile:compute)

* ***Introspection looks up the overcloud node's resources and overrides these** value. You can leave default values and Apex will get the correct values when it runs introspection on the nodes.*

** ***If capabilities profile is not specified then Apex will select node's roles** in the OPNFV cluster in a non-deterministic fashion.*

11.3 Creating the Settings Files

Edit the 2 settings files in /etc/opnfv-apex/. These files have comments to help you customize them.

1. deploy_settings.yaml This file includes basic configuration options deployment. Alternatively, there are pre-built deploy_settings files available in (`/etc/opnfv-apex/`). These files are named with the naming convention os-sdn_controller-enabled_feature-[no]ha.yaml. These files can be used in place of the (`/etc/opnfv-apex/deploy_settings.yaml`) file if one suites your deployment needs. If a pre-built deploy_settings file is choosen there is no need to customize (`/etc/opnfv-apex/deploy_settings.yaml`). The pre-built file can be used in place of the (`/etc/opnfv-apex/deploy_settings.yaml`) file.
2. network_settings.yaml This file provides Apex with the networking information that satisfies the prerequisite Network Requirements. These are specific to your environment.

11.4 Running `opnfv-deploy`

You are now ready to deploy OPNFV using Apex! `opnfv-deploy` will use the inventory and settings files to deploy OPNFV.

Follow the steps below to execute:

1. Execute opnfv-deploy `sudo opnfv-deploy [ --flat ] -n network_settings.yaml -i inventory.yaml -d deploy_settings.yaml` If you need more information about the options that can be passed to opnfv-deploy use `opnfv-deploy --help` –flat collapses all networks to a single nic, only uses the admin network from the network settings file. -n network_settings.yaml allows you to customize your networking topology.
2. Wait while deployment is executed. If something goes wrong during this part of the process, start by reviewing your network or the information in your configuration files. It's not uncommon for something small to be overlooked or mis-typed. You will also notice outputs in your shell as the deployment progresses.
3. When the deployment is complete the undercloud IP and ovecloud dashboard url will be printed. OPNFV has now been deployed using Apex.

CHAPTER

TWELVE

INSTALLATION HIGH-LEVEL OVERVIEW - VIRTUAL DEPLOYMENT

The VM nodes deployment operates almost the same way as the bare metal deployment with a few differences. `opnfv-deploy` still deploys an undercloud VM. In addition to the undercloud VM a collection of VMs (3 control nodes + 2 compute for an HA deployment or 1 control node and 1 or more compute nodes for a non-HA Deployment) will be defined for the target OPNFV deployment. The part of the toolchain that executes IPMI power instructions calls into libvirt instead of the IPMI interfaces on baremetal servers to operate the power managment. These VMs are then provisioned with the same disk images and configuration that baremetal would be. To Triple-O these nodes look like they have just built and registered the same way as bare metal nodes, the main difference is the use of a libvirt driver for the power management. Finally, the default network_settings file will deploy without modification. Customizations are welcome but not needed if a generic set of network_settings are acceptable.

CHAPTER

THIRTEEN

INSTALLATION GUIDE - VIRTUAL DEPLOYMENT

This section goes step-by-step on how to correctly install and provision the OPNFV target system to VM nodes.

13.1 Install Jumphost

Follow the instructions in the Install Bare Metal Jumphost section.

13.2 Running `opnfv-deploy`

You are now ready to deploy OPNFV! `opnfv-deploy` has virtual deployment capability that includes all of the configuration nessesary to deploy OPNFV with no modifications.

If no modifications are made to the included configurations the target environment will deploy with the following architecture:

- 1 undercloud VM
- The option of 3 control and 2 or more compute VMs (HA Deploy / default) or 1 control and 1 or more compute VM (Non-HA deploy / pass -n)
- 1-5 networks: provisioning, private tenant networking, external, storage and internal API. The API, storage and tenant networking networks can be collapsed onto the provisioning network.

Follow the steps below to execute:

1. `sudo opnfv-deploy -v [ --virtual-computes n ] [ --virtual-cpus n ] [ --virtual-ram n ] [ --flat ] -n network_settings.yaml -i inventory.yaml -d deploy_settings.yaml`
2. It will take approximately 45 minutes to an hour to stand up undercloud, define the target virtual machines, configure the deployment and execute the deployment. You will notice different outputs in your shell.
3. When the deployment is complete the IP for the undercloud and a url for the OpenStack dashboard will be displayed

13.3 Verifying the Setup - VMs

To verify the set you can follow the instructions in the Verifying the Setup section.

CHAPTER

FOURTEEN

VERIFYING THE SETUP

Once the deployment has finished, the OPNFV deployment can be accessed via the undercloud node. From the jump host ssh to the undercloud host and become the stack user. Alternativly ssh keys have been setup such that the root user on the jump host can ssh to undercloud directly as the stack user. For convenience a utility script has been provided to look up the undercloud's ip address and ssh to the undercloud all in one command. An optional user name can be passed to indicate whether to connect as the stack or root user. The stack user is default if a username is not specified.

```
opnfv-util undercloud root
su - stack
```

Once connected to undercloud as the stack user look for two keystone files that can be used to interact with the undercloud and the overcloud. Source the appropriate RC file to interact with the respective OpenStack deployment.

`source stackrc` (undercloud)
`source overcloudrc` (overcloud / OPNFV)

The contents of these files include the credentials for the administrative user for undercloud and OPNFV respectivly. At this point both undercloud and OPNFV can be interacted with just as any OpenStack installation can be. Start by listing the nodes in the undercloud that were used to deploy the overcloud.

```
source stackrc
openstack server list
```

The control and compute nodes will be listed in the output of this server list command. The IP addresses that are listed are the control plane addresses that were used to provision the nodes. Use these IP addresses to connect to these nodes. Initial authentication requires using the user heat-admin.

```
ssh heat-admin@192.0.2.7
```

To begin creating users, images, networks, servers, etc in OPNFV source the overcloudrc file or retrieve the admin user's credentials from the overcloudrc file and connect to the web Dashboard.

You are now able to follow the *OpenStack Verification* section.

CHAPTER

FIFTEEN

OPENSTACK VERIFICATION

Once connected to the OPNFV Dashboard make sure the OPNFV target system is working correctly:

1. In the left pane, click Compute -> Images, click Create Image.
2. Insert a name “cirros”, Insert an Image Location `http://download.cirros-cloud.net/0.3.4/cirros-0.3.4-x8`
3. Select format “QCOW2”, select Public, then click Create Image.
4. Now click Project -> Network -> Networks, click Create Network.
5. Enter a name “internal”, click Next.
6. Enter a subnet name “internal_subnet”, and enter Network Address `172.16.1.0/24`, click Next.
7. Now go to Project -> Compute -> Instances, click Launch Instance.
8. Enter Instance Name “first_instance”, select Instance Boot Source “Boot from image”, and then select Image Name “cirros”.
9. Click Launch, status will cycle though a couple states before becoming “Active”.
10. Steps 7 though 9 can be repeated to launch more instances.
11. Once an instance becomes “Active” their IP addresses will display on the Instances page.
12. Click the name of an instance, then the “Console” tab and login as “cirros”/”cubswin:)”
13. To verify storage is working, click Project -> Compute -> Volumes, Create Volume
14. Give the volume a name and a size of 1 GB
15. Once the volume becomes “Available” click the dropdown arrow and attach it to an instance.

Congratulations you have successfully installed OPNFV!

CHAPTER

SIXTEEN

FREQUENTLY ASKED QUESTIONS

CHAPTER

SEVENTEEN

LICENSE

All Apex and “common” entities are protected by the Apache 2.0 License.

CHAPTER

EIGHTEEN

REFERENCES

18.1 OPNFV

OPNFV Home Page

OPNFV Genesis project page

OPNFV Apex project page

18.2 OpenStack

OpenStack Mitaka Release artifacts

OpenStack documentation

18.3 OpenDaylight

Upstream OpenDaylight provides a number of packaging and deployment options meant for consumption by downstream projects like OPNFV.

Currently, OPNFV Apex uses OpenDaylight's Puppet module, which in turn depends on OpenDaylight's RPM.

18.4 RDO Project

RDO Project website

Authors Tim Rozet (trozet@redhat.com)

Authors Dan Radez (dradez@redhat.com)

Version 3.0

CHAPTER

NINETEEN

INDICES AND TABLES

- search