Install OPNFV on IPv6-Only Infrastructure

Release draft (7c6658f)

OPNFV

August 22, 2016

This section provides instructions to install OPNFV on IPv6-only Infrastructure. All underlay networks and API endpoints will be IPv6-only except:

1. "admin" network in underlay/undercloud still has to be IPv4, due to lack of support of IPMI over IPv6 or PXE over IPv6.
2. OVS VxLAN (or GRE) tunnel endpoint is still IPv4 only, although IPv6 traffic can be encapsulated within the tunnel.
3. Metadata server is still IPv4 only.

Except the limitations above, the use case scenario of the IPv6-only infrastructure includes:

1. Support OPNFV deployment on an IPv6 only infrastructure.
2. Horizon/ODL-DLUX access using IPv6 address from an external host.
3. OpenStack API access using IPv6 addresses from various python-clients.
4. Ability to create Neutron Routers, IPv6 subnets (e.g. SLAAC/DHCPv6-Stateful/ DHCPv6-Stateless) to support North-South traffic.
5. Inter VM communication (East-West routing) when VMs are spread across two compute nodes.
6. VNC access into a VM using IPv6 addresses.

CHAPTER

ONE

INSTALL OPNFV IN OPENSTACK-ONLY ENVIRONMENT

Apex Installer:

```
# HA, Virtual deployment in OpenStack-only environment
./opnfv-deploy -v -d /etc/opnfv-apex/os-nosdn-nofeature-ha.yaml \
-n /etc/opnfv-apex/network_setting_v6.yaml

# HA, Bare Metal deployment in OpenStack-only environment
./opnfv-deploy -d /etc/opnfv-apex/os-nosdn-nofeature-ha.yaml \
-i <inventory file> -n /etc/opnfv-apex/network_setting_v6.yaml

# Non-HA, Virtual deployment in OpenStack-only environment
./opnfv-deploy -v -d /etc/opnfv-apex/os-nosdn-nofeature-noha.yaml \
-n /etc/opnfv-apex/network_setting_v6.yaml

# Non-HA, Bare Metal deployment in OpenStack-only environment
./opnfv-deploy -d /etc/opnfv-apex/os-nosdn-nofeature-noha.yaml \
-i <inventory file> -n /etc/opnfv-apex/network_setting_v6.yaml

# Note:
#
# 1. Parameter ""-v" is mandatory for Virtual deployment
# 2. Parameter "-i <inventory file>" is mandatory for Bare Metal deployment
# 2.1 Refer to https://git.opnfv.org/cgit/apex/tree/config/inventory for examples of inventory file
# 3. You can use "-n /etc/opnfv-apex/network_setting.yaml" for deployment in IPv4 infrastructure
```

Joid Installer:

```
# HA deployment in OpenStack-only environment
./deploy.sh -o mitaka -s nosdn -t ha -l default -f ipv6

# Non-HA deployment in OpenStack-only environment
./deploy.sh -o mitaka -s nosdn -t nonha -l default -f ipv6
```

Please **NOTE** that:

- You need to refer to **installer's documentation** for other necessary parameters applicable to your deployment.
- You need to refer to **Release Notes** and **installer's documentation** if there is any issue in installation.

CHAPTER

TWO

INSTALL OPNFV IN OPENSTACK WITH ODL-L2 ENVIRONMENT

Apex Installer:

```
# HA, Virtual deployment in OpenStack with Open Daylight L2-only environment
./opnfv-deploy -v -d /etc/opnfv-apex/os-odl_l2-nofeature-ha.yaml \
-n /etc/opnfv-apex/network_setting_v6.yaml

# HA, Bare Metal deployment in OpenStack with Open Daylight L2-only environment
./opnfv-deploy -d /etc/opnfv-apex/os-odl_l2-nofeature-ha.yaml \
-i <inventory file> -n /etc/opnfv-apex/network_setting_v6.yaml

# Non-HA deployment in OpenStack with Open Daylight L2-only environment
# There is no settings file provided by default for odl_l2 non-HA deployment
# You need to copy /etc/opnfv-apex/os-odl_l2-nofeature-ha.yaml to another file
# e.g. /etc/opnfv-apex/os-odl_l2-nofeature-noha.yaml
# and change the "ha_enabled" parameter to be "false", i.e.: "ha_enabled: false", and:

# - For Non-HA, Virtual deployment
./opnfv-deploy -v -d /etc/opnfv-apex/os-odl_l2-nofeature-noha.yaml \
-n /etc/opnfv-apex/network_setting_v6.yaml

# - For Non-HA, Bare Metal deployment
./opnfv-deploy -d /etc/opnfv-apex/os-odl_l2-nofeature-noha.yaml \
-i <inventory file> -n /etc/opnfv-apex/network_setting_v6.yaml

# Note:
#
# 1. Parameter ""-v" is mandatory for Virtual deployment
# 2. Parameter "-i <inventory file>" is mandatory for Bare Metal deployment
# 2.1 Refer to https://git.opnfv.org/cgit/apex/tree/config/inventory for examples of inventory file
# 3. You can use "-n /etc/opnfv-apex/network_setting.yaml" for deployment in IPv4 infrastructure
```

Joid Installer:

```
# HA deployment in OpenStack with Open Daylight L2-only environment
./deploy.sh -o mitaka -s odl -t ha -l default -f ipv6

# Non-HA deployment in OpenStack with Open Daylight L2-only environment
./deploy.sh -o mitaka -s odl -t nonha -l default -f ipv6
```

Please **NOTE** that:

- You need to refer to **installer's documentation** for other necessary parameters applicable to your deployment.
- You need to refer to **Release Notes** and **installer's documentation** if there is any issue in installation.

CHAPTER

THREE

TESTING METHODOLOGY

There are 2 levels of testing to validate the deployment.

3.1 Underlay Testing

Underlay Testing is to validate that API endpoints are listening on IPv6 addresses. This can be as simple as validating Keystone service, and as complete as validating each API endpoint. It is important to reuse Tempest API testing.

Please **Note** that, to the best of our knowledge, Tempest API testing does not validate API endpoints listening on IPv6 addresses. Thus Underlay Testing is postponed to future release until Tempest API testing is ready to validate API endpoints listening on IPv6 addresses.

3.2 Overlay Testing

Overlay Testing is to validate that IPv6 is supported in tenant networks, subnets and routers. Both Tempest API testing and Tempest Scenario testing are used in our Overlay Testing.

Tempest API testing validates that the Neutron API supports the creation of IPv6 networks, subnets, routers, etc:

```
tempest.api.network.test_networks.BulkNetworkOpsIpV6Test.test_bulk_create_delete_network
tempest.api.network.test_networks.BulkNetworkOpsIpV6Test.test_bulk_create_delete_port
tempest.api.network.test_networks.BulkNetworkOpsIpV6Test.test_bulk_create_delete_subnet
tempest.api.network.test_networks.NetworksIpV6Test.test_create_update_delete_network_subnet
tempest.api.network.test_networks.NetworksIpV6Test.test_external_network_visibility
tempest.api.network.test_networks.NetworksIpV6Test.test_list_networks
tempest.api.network.test_networks.NetworksIpV6Test.test_list_subnets
tempest.api.network.test_networks.NetworksIpV6Test.test_show_network
tempest.api.network.test_networks.NetworksIpV6Test.test_show_subnet
tempest.api.network.test_networks.NetworksIpV6TestAttrs.test_create_update_delete_network_subnet
tempest.api.network.test_networks.NetworksIpV6TestAttrs.test_external_network_visibility
tempest.api.network.test_networks.NetworksIpV6TestAttrs.test_list_networks
tempest.api.network.test_networks.NetworksIpV6TestAttrs.test_list_subnets
tempest.api.network.test_networks.NetworksIpV6TestAttrs.test_show_network
tempest.api.network.test_networks.NetworksIpV6TestAttrs.test_show_subnet
tempest.api.network.test_ports.PortsIpV6TestJSON.test_create_port_in_allowed_allocation_pools
tempest.api.network.test_ports.PortsIpV6TestJSON.test_create_port_with_no_securitygroups
tempest.api.network.test_ports.PortsIpV6TestJSON.test_create_update_delete_port
tempest.api.network.test_ports.PortsIpV6TestJSON.test_list_ports
tempest.api.network.test_ports.PortsIpV6TestJSON.test_show_port
tempest.api.network.test_routers.RoutersIpV6Test.test_add_multiple_router_interfaces
tempest.api.network.test_routers.RoutersIpV6Test.test_add_remove_router_interface_with_port_id
tempest.api.network.test_routers.RoutersIpV6Test.test_add_remove_router_interface_with_subnet_id
```

```
tempest.api.network.test_routers.RoutersIpV6Test.test_create_show_list_update_delete_router
tempest.api.network.test_security_groups.SecGroupIPv6Test.test_create_list_update_show_delete_securit
tempest.api.network.test_security_groups.SecGroupIPv6Test.test_create_show_delete_security_group_rule
tempest.api.network.test_security_groups.SecGroupIPv6Test.test_list_security_groups
```

Tempest Scenario testing validates some specific overlay IPv6 scenarios (i.e. use cases) as follows:

```
tempest.scenario.test_network_v6.TestGettingAddress.test_dhcp6_stateless_from_os
tempest.scenario.test_network_v6.TestGettingAddress.test_dualnet_dhcp6_stateless_from_os
tempest.scenario.test_network_v6.TestGettingAddress.test_dualnet_multi_prefix_dhcpv6_stateless
tempest.scenario.test_network_v6.TestGettingAddress.test_dualnet_multi_prefix_slaac
tempest.scenario.test_network_v6.TestGettingAddress.test_dualnet_slaac_from_os
tempest.scenario.test_network_v6.TestGettingAddress.test_multi_prefix_dhcpv6_stateless
tempest.scenario.test_network_v6.TestGettingAddress.test_multi_prefix_slaac
tempest.scenario.test_network_v6.TestGettingAddress.test_slaac_from_os
```

The above Tempest API testing and Scenario testing are quite comprehensive to validate overlay IPv6 tenant networks. They are part of OpenStack default Smoke Tests, run in FuncTest and integrated into OPNFV's CI/CD environment.