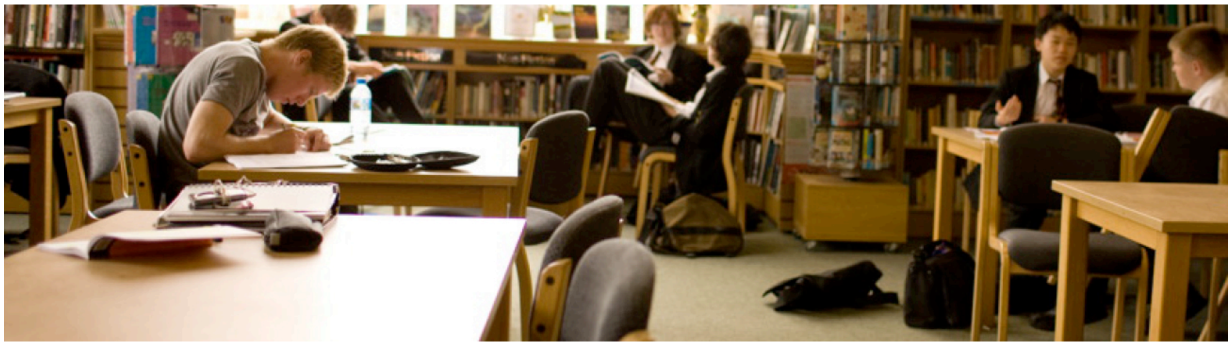
# IPv6 support in OpenStack Neutron

Technical Brief
Last Update: 05/04/15



OpenStack Neutron has made and continues to make a lot of progress to enable IPv6 tenant networks. From supporting various IPv6 addressing schemes for tenant networks to enabling features that support and use IPv6 capabilities, the IPv6 footprint in Neutron is expanding. In this technical brief, we'll cover the various IPv6 Neutron changes that have either been already merged into the Neutron Juno and Kilo release or are undergoing discussions as part of the Liberty development release cycle. The reader will also be able to understand the different IPv6 deployment use cases that are possible with Neutron.

## Terminology

Tenant Networks and Routers – Neutron resources that are provisioned by the tenant and available only to the OpenStack tenant for network connectivity.

Provider Network and Routers – Neutron resources that are provisioned by the OpenStack administrator and also provide access to existing network infrastructure in the data center for tenant network connectivity.

External Network and Routers – External Network resource that is provisioned by the OpenStack administrator to enable outside access for tenant networks and also provide connectivity to external routers in the Data Center configured outside of OpenStack.

## IPv6 addressing support in Neutron

IPv6 supports three different addressing schemes for address configuration and for providing optional network information.

- Stateless Address Auto Configuration (SLAAC) – Address configuration using Router Advertisement (RA)
- DHCPv6-stateless – Address configuration using RA and optional information using DHCPv6
- DHCPv6-stateful – Address configuration and optional information using DHCPv6

Tenant can either configure Neutron or rely on external routers and services to provide RA, DHCPv6 address and optional information for their networks. There are two Neutron subnet attributes - ipv6_ra_mode and ipv6_address_mode – that determine how IPv6 addressing and network information is provided to tenant instances.

- ipv6_ra_mode – Determines who sends RA.
- ipv6_address_mode – Determines how instances obtain IPv6 address, default gateway, and/or optional information.

For the above two attributes to be effective, enable_dhcp must be set to True in Neutron. With enable_dhcp set to True, if neither attribute is configured, it is considered to be a case of DHCPv6-stateful.

## SLAAC

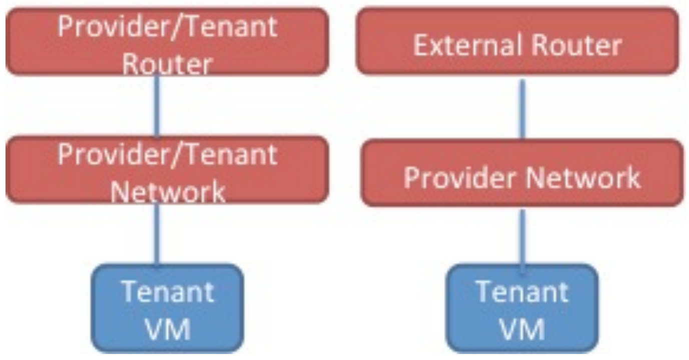
For SLAAC, the possible combinations for the attributes are -

| **ipv6_ra_mode** | **ipv6_address_mode** | **Result** |
|---|---|---|
| SLAAC | Not specified | Address using Neutron router |
| Not specified | SLAAC | Address using external router |
| SLAAC | SLAAC | Address using Neutron router |

Setting SLAAC for ipv6_ra_mode configures Neutron router with radvd agent to send RA. This results in the following values set for the address configuration flags in the RA messages.

| Address Configuration Flags | Value |
|---|---|
| Auto | 1 |
| Managed | 0 |
| Other | 0 |

Deployment scenarios with SLAAC are captured below.



The following blueprints were implemented in the Juno development release cycle to achieve the above –

- https://blueprints.launchpad.net/neutron/+spec/ipv6-two-attributes
- https://blueprints.launchpad.net/neutron/+spec/ipv6-provider-nets-slaac
- https://blueprints.launchpad.net/neutron/+spec/neutron-ipv6-radvd-ra (Cisco lead)

## DHCPv6-stateless

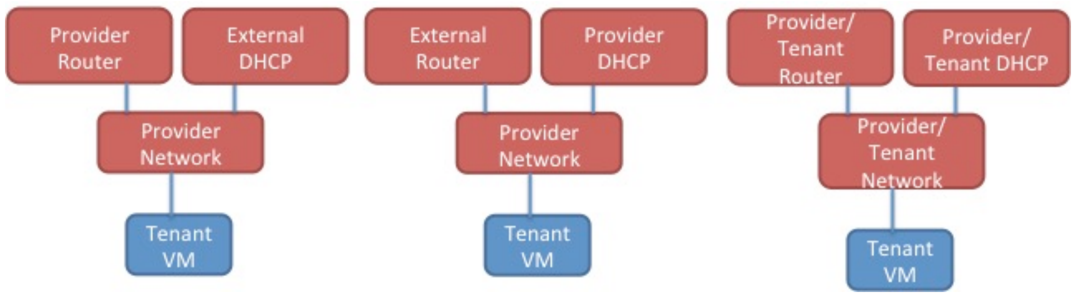
For DHCPv6-stateless, the possible combinations are –

| **ipv6_ra_mode** | **ipv6_address_mode** | **Result** |
|---|---|---|
| DHCPv6-stateless | Not specified | Address using Neutron router and optional information using external DHCP service |
| Not specified | DHCPv6-stateless | Address using external router and optional information using Neutron DHCP implementation |
| DHCPv6-stateless | DHCPv6-stateless | Address and optional information using Neutron router and DHCP implementation respectively |

Setting DHCPv6-stateless for ipv6_ra_mode configures Neutron router with radvd agent to send RA. The table below captures the values set for the address configuration flags in the RA messages in this scenario. Similarly, setting DHCPv6-stateless for ipv6_address_mode configures Neutron DHCP implementation to provide the additional network information.

| Address Configuration Flags | Value |
|---|---|
| Auto | 1 |
| Managed | 0 |
| Other | 1 |

Deployment scenarios are captured below.



This Juno blueprint implementation captures the details –

- https://blueprints.launchpad.net/neutron/+spec/dnsmasq-ipv6-dhcpv6-stateless

## DHCPv6-stateful

For DHCPv6-stateful, the possible combinations are -

| **ipv6_ra_mode** | **ipv6_address_mode** | **Result** |
|---|---|---|
| DHCPv6-stateful | Not specified | Address and optional information using external DHCP service |
| Not specified | DHCPv6-stateful | Address and optional information using Neutron DHCP implementation |
| DHCPv6-stateful | DHCPv6-stateful | Address and optional information using Neutron DHCP implementation |

Setting DHCPv6-stateful for ipv6_ra_mode results in the configuration of Auto, Managed and Others flag in the RA message as captured in the table below. Setting DHCPv6-statelful for ipv6_address_mode configures Neutron DHCP implementation to provide the address and optional information.  The above diagram also captures the deployment scenarios for DHCPv6-stateful.

| Address Configuration Flags | Value |
|---|---|
| Auto | 0 |
| Managed | 1 |
| Other | 1 |

This Juno blueprint implementation captures the details –

- https://blueprints.launchpad.net/neutron/+spec/dnsmasq-ipv6-dhcpv6-stateful

Also, to enable scalable and production IPv6 deployments with stateful addressing and for IPv4 address management as well, Cisco OpenStack team is working on Cisco Prime Network Registrar integration to provide alternate to the reference DHCP implementation in Neutron that uses dnsmasq.

## Security fixes to support IPv6 addressing

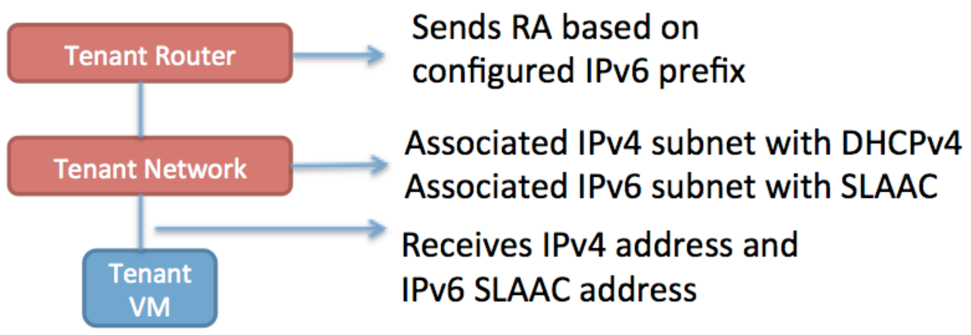
From an instance network security point of view, to support IPv6 addressing, a few important patches were submitted in security group implementation. This included enabling passing certain ICMPv6 messages by default and permitting ICMPv6 RA messages from known routers.  Also, traffic hairpinning to permit communication of instances from private IPv4 to floating IPv4 address had to be disabled for IPv6. This was needed to handle IPv6 Duplicate Address Detection (DAD) multicast messages in SLAAC addressing mode.

These fixes are already available as part of the Juno release.

## Neutron Port behavior change for IPv6 addressing

### Network Ports

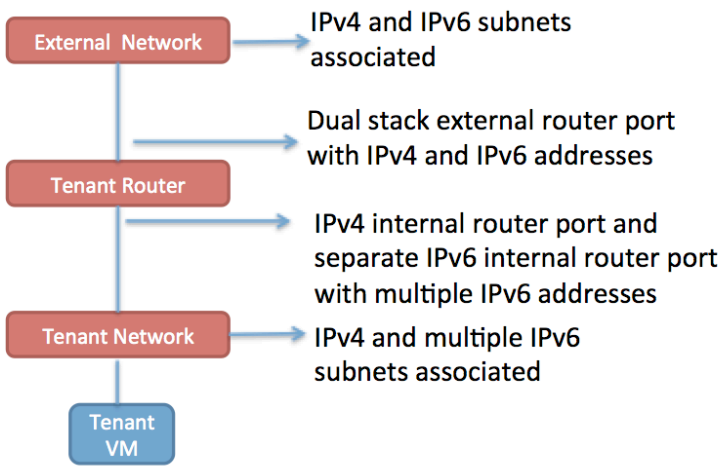
In order to support both IPv4 and IPv6 addressing for various ports in Neutron, port behavior changes have been introduced. New or existing Neutron networks that get associated with a SLAAC enabled IPv6 subnet result in all Neutron ports within the network to get an IPv6 address assigned automatically. This is because when RA multicast messages are sent out on a Neutron network, they are received by all IPv6 capable interfaces on the network and configure an IPv6 address. The capability to reflect the IPv6 SLAAC address in addition to other IPv4 address on the network port has been added.


Tenant Router
Sends RA based on configured IPv6 prefix
Tenant Network
Associated IPv4 subnet with DHCPv4
Associated IPv6 subnet with SLAAC
Tenant VM
Receives IPv4 address and IPv6 SLAAC address

Additionally, in case of IPv4, subnet delete was not permitted when ports existed with an IPv4 address assigned from that subnet. However, with IPv6 SLAAC addressing, tenant networks that no longer want to configure IPv6 address based on RA from the tenant router, should be able to delete the IPv6 subnet even though ports exist that have an IPv6 address configured from the SLAAC subnet. This IPv6 subnet delete operation results in clearing of the IPv6 SLAAC address from the existing port and new ports on the tenant network no longer get configured automatically with a IPv6 SLAAC address.

### Router ports

Internal and external router port behavior has changed as well to accommodate the IPv6 addressing. Internal router ports, that act as default gateway ports for a network, will share a common port for all IPv6 subnets associated with the network.  This implies that there will be an IPv6 internal router interface with multiple IPv6 addresses from each of the IPv6 subnets associated with the network and a separate IPv4 internal router interface for the IPv4 subnet. On the other hand, external router ports can now have a dual-stack configuration with an IPv4 and an IPv6 address assigned to them.


External Network
IPv4 and IPv6 subnets associated
Dual stack external router port with IPv4 and IPv6 addresses
Tenant Router
IPv4 internal router port and separate IPv6 internal router port with multiple IPv6 addresses
Tenant Network
IPv4 and multiple IPv6 subnets associated
Tenant VM

The following blueprint was implemented as part of the Kilo development cycle –

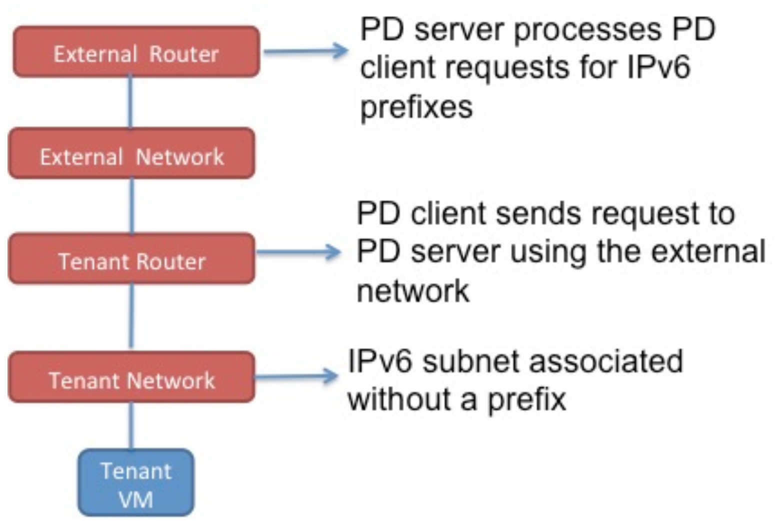
- https://blueprints.launchpad.net/neutron/+spec/multiple-ipv6-prefixes (Cisco lead)

## Handling IPv6 Global Unique Addresses (GUA) for Neutron networks

### IPv6 Prefix Delegation (PD) in Neutron

IPv6 prefixes that use Global Unicast Address (GUA) format generate globally unique routable addresses. For IPv4 Neutron networks, tenants provide IPv4 CIDR when creating subnets and these addresses can overlap across tenants. However, incase of IPv6 subnets that use GUA format, prefixes and addresses among tenants cannot overlap, as they are unique.

To simplify the process of IPv6 GUA subnet prefix selection for tenant networks and to avoid any conflict among tenants for IPv6 global addresses, IPv6 Prefix Delegation (PD) mechanism is being discussed in the Neutron community. IPv6 PD provides a mechanism to automatically configure IPv6 prefixes and addresses on routers and hosts, and can be used in Neutron so that tenants can determine their address based on the prefix provided by an external service. The current spec proposal design has a prefix delegating router (or a PD server outside of Neutron) that is configured with a set of prefixes. A PD process will start when a Neutron router acting as a PD requester (or PD client) requests configuration information through DHCPv6 messages. When the PD server receives the request, it will then select an available prefix for delegation to the PD client.


External Router
PD server processes PD client requests for IPv6 prefixes
External Network
Tenant Router
PD client sends request to PD server using the external network
Tenant Network
IPv6 subnet associated without a prefix
Tenant VM

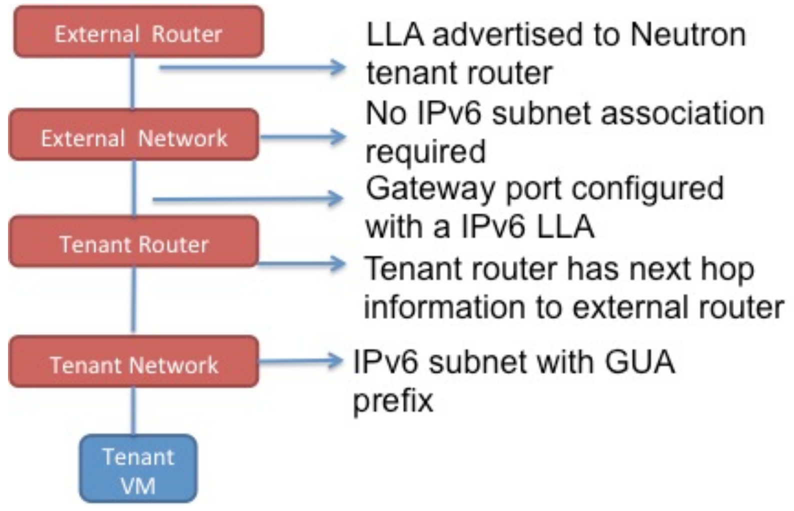
The following blueprint is being developed and reviewed as part of the Liberty development cycle –

- https://blueprints.launchpad.net/neutron/+spec/ipv6-prefix-delegation (Cisco lead)

## IPv6 Routing for Neutron Networks

Neutron tenant networks that are assigned GUA prefixes and addresses don't require NAT on the Neutron router external gateway port to access the outside world.  This implies, as compared to the IPv4 implementation in Neutron, a IPv6 external gateway port doesn't require a GUA and that a GUA IPv6 subnet prefix is not necessarily needed for the Neutron external network. By default, a IPv6 LLA associated with the external gateway port can be used for routing purposes.

To handle the above IPv6 routing capability, the implementation of router-gateway-set API in Neutron is modified so that an IPv6 subnet is not required for the external network that is associated with the Neutron router. Additionally, two mechanisms are available to support explicit next hop configuration on the Neutron router. In the absence of an upstream RA, ipv6_gateway flag can be set with the external router gateway LLA in the Neutron L3 agent configuration file. If the RA is enabled, the external gateway port can learn the LLA address of the upstream router. This will enable instances that have a GUA address to route traffic to the external network without any NAT.


External Router
LLA advertised to Neutron tenant router
External Network
No IPv6 subnet association required
Gateway port configured with a IPv6 LLA
Tenant Router
Tenant router has next hop information to external router
Tenant Network
IPv6 subnet with GUA prefix
Tenant VM

The following blueprint was implemented as part of the Kilo development cycle –

- https://blueprints.launchpad.net/neutron/+spec/ipv6-router (Cisco lead)

## Summary

As OpenStack deployments increase in size and those that require advance networking capabilities, adoption of Neutron and IPv6 will continue to grow. The Neutron community and developers from various companies including Comcast, Cisco, IBM, RedHat, Nephos6 and others have been active and instrumental in developing and reviewing IPv6 code in Neutron. If you would like to contribute, either send a note to the OpenStack dev mailing list or join the weekly Neutron IPv6 meeting to learn more.