\Orchestrating a brighter world NEC

# Lessons in IPv6 deployment in OpenStack environments

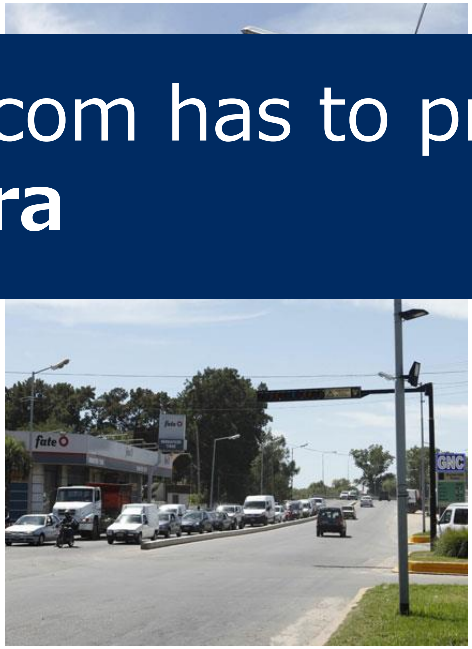
May 8, 2017

Akihiro Motoki
OSS Promotion Center, NEC

# Why IPv6?

**IPv4 Address has run out!**

**Smart City and IoT needs lots of addresses**



All the Industry and Telecom has to prepare **IPv6 Era**

# Agenda

- Many consideration points on IPv6 deployments with OpenStack

- [0] Understanding IPv6 words
- [1] IPv6 address allocation
  - How to manage IPv6 address pools
  - Global unique addresses for tenants
- [2] IPv6 address configuration
  - How to distribute IPv6 address to clients
- [3] Routing considerations
  - How to reach tenants’ network?
  - How to prevent unauthorized global addresses?
- [4] VNF choices
  - Neutron router?
  - 3-rd party VNF router? Firewall?

- This presentation is based on OpenStack Mitaka release

# IPv6 address allocation

How to manage IPv6 address pools?

# IPv6 address allocation

- What we need?

- GUA (Global unique address) are assigned to tenants
- GUA should be assigned to a specified address range

→ **Neutron Subnet Pool**

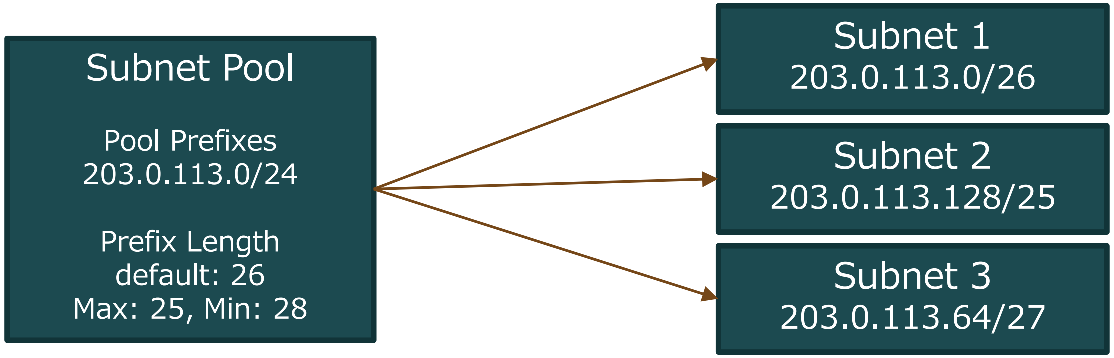
- Tenants still need to use their own addresses
  - Unless they are not connected to external
  - ULA (Unique local address)

→ Legacy way to specify CIDR
(or tenant subnet pool)

# Neutron "subnet pool"

- "Subnet pool" defines CIDR pool and allocates CIDR for a subnet from the pool
- What we can?
  - No need to care which CIDR is used or not. Subnet pool manages assigned CIDR.
  - OpenStack operators pre-defines a desired IP address range
  - It allows us to assign non-overlapping address ranges across tenants

```
neutron subnetpool-create
   --pool-prefix 203.0.113.0/24 --default-prefixlen 26
   --shared --is-default demo-subnetpool
neutron subnet-create --subnetpool demo-subnetpool --ip-version 4 demo-net
neutron subnet-create --use-default-subnetpool --ip-version 4 demo-net
neutron subnetpool-update
--pool-prefix 203.0.113.0/24 --pool-prefix 198.51.100.0/24 demo-subnetpool
```



# IPv6 address allocation – admin

## We use “subnet pool” to manage IPv6 address ranges

## Create a subnet pool for GUA

- openstack subnet pool create
  --pool-prefix 2001:DB8:1234::/48
  --default-prefix-length 64 --min-prefix-length 64 --max-prefix-length 64
  subnet-pool-ipv6
- Specify the global address range as a pool prefix
- /64 is suggested as prefix length as most IPv6 routers assume 64 in RA

## Mark it as “shared” so that tenants can consume it

- openstack subnet pool set --share subnet-pool-ipv6

## (optional) Mark it as a default subnet pool

- openstack subnet pool set --default subnet-pool-ipv6
- Tenants do not need to specify the subnet pool name when creating a subnet

# IPv6 address allocation - tenant

## Check per-defined subnet pools

- openstack subnet pool list

## Create a subnet using the subnet pool

- openstack subnet create --use-default-subnet-pool --ip-version 6 --ipv6-ra-mode XXX –ipv6-address-mode XXX --network net1 subnet1
- You can specify a subnet pool explicitly but using a default pool would be easier

## Quota on subnet pool

- Subnet pool supports quota
- In IPv6 case, it is calculated as the number of /64 subnets.
- If the subnet pool quota is 3, a tenant can allocate 3 /64 subnets.

## Tenants still can specify their own address range

- Limited to local use and no internet connection

# IPv6 address configuration

# IPv6 address configuration modes

## Three configuration modes:

- SLLAC, DHCPv6 stateless, DHCPv6 stateful

## SLACC

- IPv6 address of a client is configured based on RA (Router advertisement)
- Gateway is also configured.
- Optionally, DNS information(if RFC6106), MTU and so on can be configured.
- Only /64 prefix is used

## DHCPv6 stateless

- IPv6 address is configured based on RA.
- Other information is retrieved via DHCPv6.
- Looks used most commonly
- Only way to distribute DNS info before RFC6106

## DHCPv6 stateful

- All information is configured based on DHCPv6
- There is information that GW is not configured properly.

# IPv6 address configuration modes

## The configuration mode is determined based on RA flag

- M (Managed) => 0 (RA/DHCPv6 stateless), 1 (DHCPv6 stateful)
- O (Other) => 0 (SLAAC), 1 (use DHCPv6)

## Need to selection address mode, depending on router implementation used

- Neutron exposes all modes, but it is not necessarily all modes are available...

# Understanding Neutron IPv6 two modes

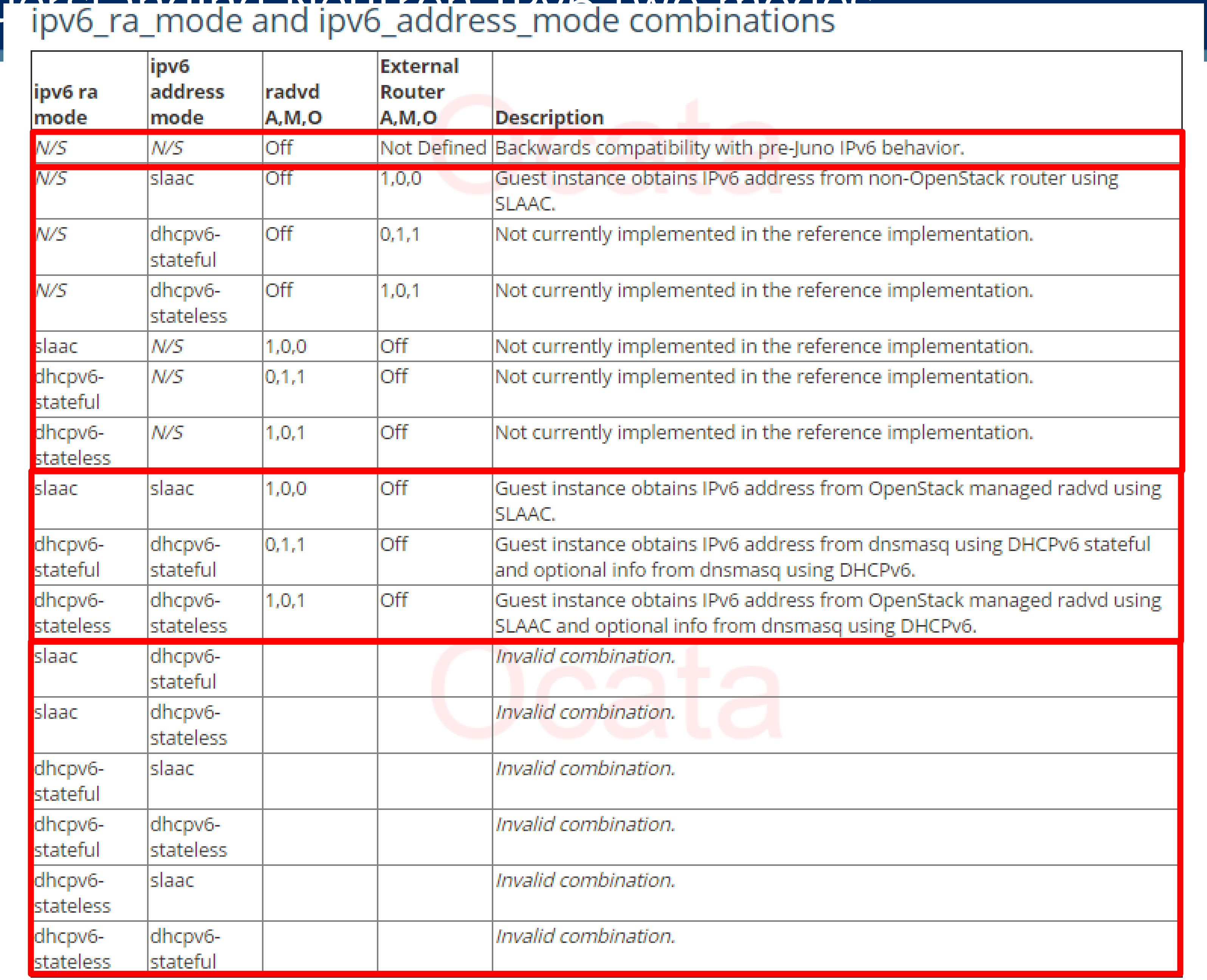
## Two attributes related to IPv6 address configuration:

- ipv6_address_mode
- ipv6_ra_mode

## There are constraints between two modes

- OpenStack networking guide
- https://docs.openstack.org/ocata/networking-guide/config-ipv6.html

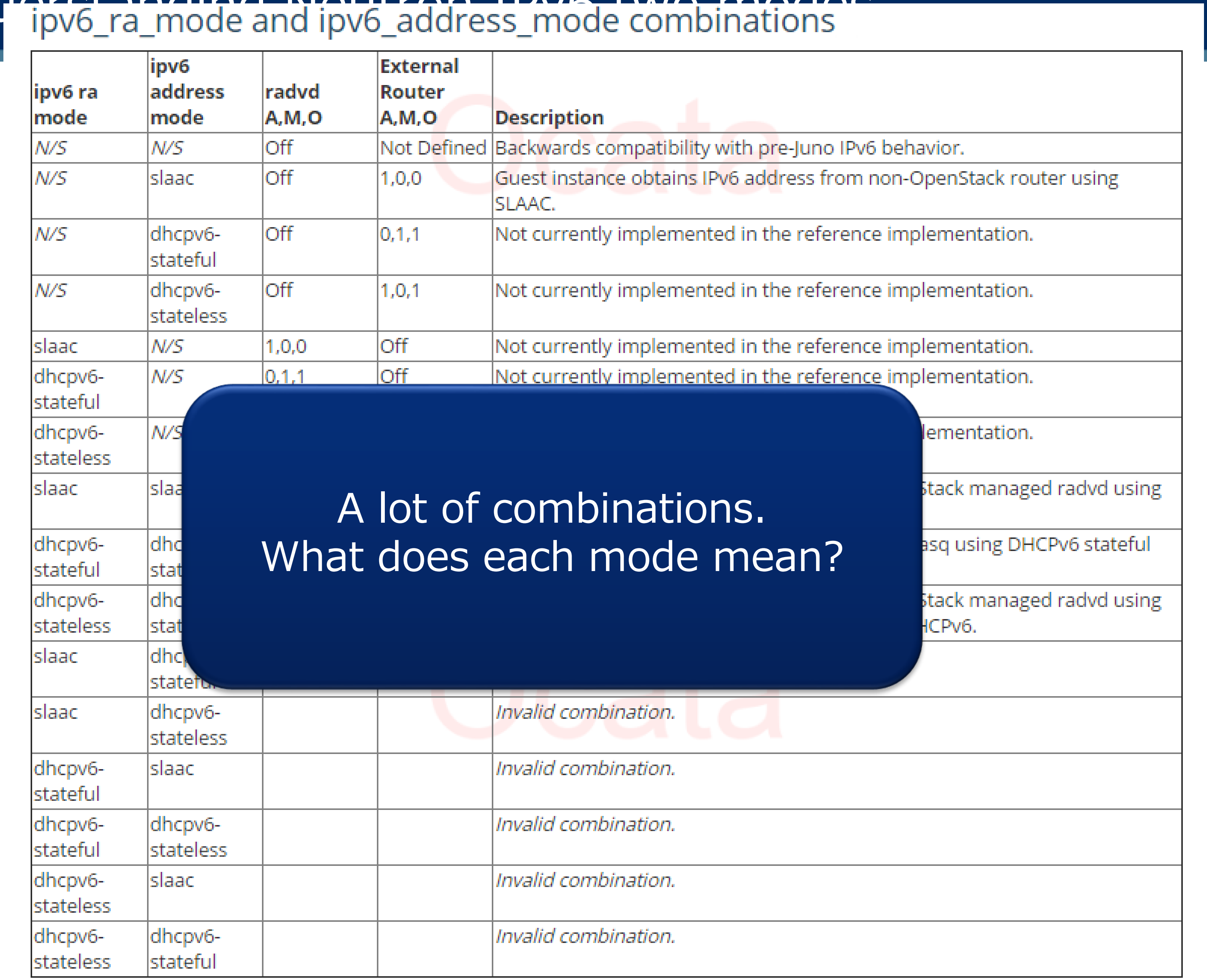
# Understanding Neutron IPv6 two modes



## ipv6_ra_mode and ipv6_address_mode combinations

| ipv6 ra mode | ipv6 address mode | radvd A,M,O | External Router A,M,O | Description |
|---|---|---|---|---|
| *N/S* | *N/S* | Off | Not Defined | Backwards compatibility with pre-Juno IPv6 behavior. |
| *N/S* | slaac | Off | 1,0,0 | Guest instance obtains IPv6 address from non-OpenStack router using SLAAC. |
| *N/S* | dhcpv6-stateful | Off | 0,1,1 | Not currently implemented in the reference implementation. |
| *N/S* | dhcpv6-stateless | Off | 1,0,1 | Not currently implemented in the reference implementation. |
| slaac | *N/S* | 1,0,0 | Off | Not currently implemented in the reference implementation. |
| dhcpv6-stateful | *N/S* | 0,1,1 | Off | Not currently implemented in the reference implementation. |
| dhcpv6-stateless | *N/S* | 1,0,1 | Off | Not currently implemented in the reference implementation. |
| slaac | slaac | 1,0,0 | Off | Guest instance obtains IPv6 address from OpenStack managed radvd using SLAAC. |
| dhcpv6-stateful | dhcpv6-stateful | 0,1,1 | Off | Guest instance obtains IPv6 address from dnsmasq using DHCPv6 stateful and optional info from dnsmasq using DHCPv6. |
| dhcpv6-stateless | dhcpv6-stateless | 1,0,1 | Off | Guest instance obtains IPv6 address from OpenStack managed radvd using SLAAC and optional info from dnsmasq using DHCPv6. |
| slaac | dhcpv6-stateful | | | *Invalid combination.* |
| slaac | dhcpv6-stateless | | | *Invalid combination.* |
| dhcpv6-stateful | slaac | | | *Invalid combination.* |
| dhcpv6-stateful | dhcpv6-stateless | | | *Invalid combination.* |
| dhcpv6-stateless | slaac | | | *Invalid combination.* |
| dhcpv6-stateless | dhcpv6-stateful | | | *Invalid combination.* |

# Understanding Neutron IPv6 two modes



## ipv6_ra_mode and ipv6_address_mode combinations

| ipv6 ra mode | ipv6 address mode | radvd A,M,O | External Router A,M,O | Description |
|---|---|---|---|---|
| *N/S* | *N/S* | Off | Not Defined | Backwards compatibility with pre-Juno IPv6 behavior. |
| *N/S* | slaac | Off | 1,0,0 | Guest instance obtains IPv6 address from non-OpenStack router using SLAAC. |
| *N/S* | dhcpv6-stateful | Off | 0,1,1 | Not currently implemented in the reference implementation. |
| *N/S* | dhcpv6-stateless | Off | 1,0,1 | Not currently implemented in the reference implementation. |
| slaac | *N/S* | 1,0,0 | Off | Not currently implemented in the reference implementation. |
| dhcpv6-stateful | *N/S* | 0,1,1 | Off | Not currently implemented in the reference implementation. |
| dhcpv6-stateless | *N/S* | | | lementation. |
| slaac | slaa | | | tack managed radvd using |
| dhcpv6-stateful | dhc<br>stat | | | asq using DHCPv6 stateful |
| dhcpv6-stateless | dhc<br>stat | | | tack managed radvd using<br>HCPv6. |
| slaac | dhc<br>statef | | | |
| slaac | dhcpv6-stateless | | | *Invalid combination.* |
| dhcpv6-stateful | slaac | | | *Invalid combination.* |
| dhcpv6-stateful | dhcpv6-stateless | | | *Invalid combination.* |
| dhcpv6-stateless | slaac | | | *Invalid combination.* |
| dhcpv6-stateless | dhcpv6-stateful | | | *Invalid combination.* |

A lot of combinations.
What does each mode mean?

# Understanding IPv6 two mode

## IPv6 address mode

- Specifies how IPv6 address is generated and assigned
- IPAM is mainly involved in it.
- Also controls if the reference DHCP implementation serves

- “slaac”
  - Neutron generates a port address based on MAC address (EUI-64)
- “dhcpv6-stateless”
  - Neutron generates a port address based on MAC address (EUI-64)
  - Neutron provides DHCP options for port
- “dhcpv6-stateful”
  - Any address can be configured.
  - Perhaps non-/64 prefix can be used
- Not Specified (N/S)
  - (Backward compatibility)
  - Any static address can be configured.

# Understanding IPv6 two mode

## IPv6 RA mode

- Specifies how neutron router sends RA

## “slaac”

## “dhcpv6-stateless”

## “dhcpv6-stateful”

- Neutron setup radvd on a router and provides RA
- RA flags are set accordingly

## Not Specified (N/S)

- Neutron does nothing.
- radvd on a router is not setup

# Deployment patterns

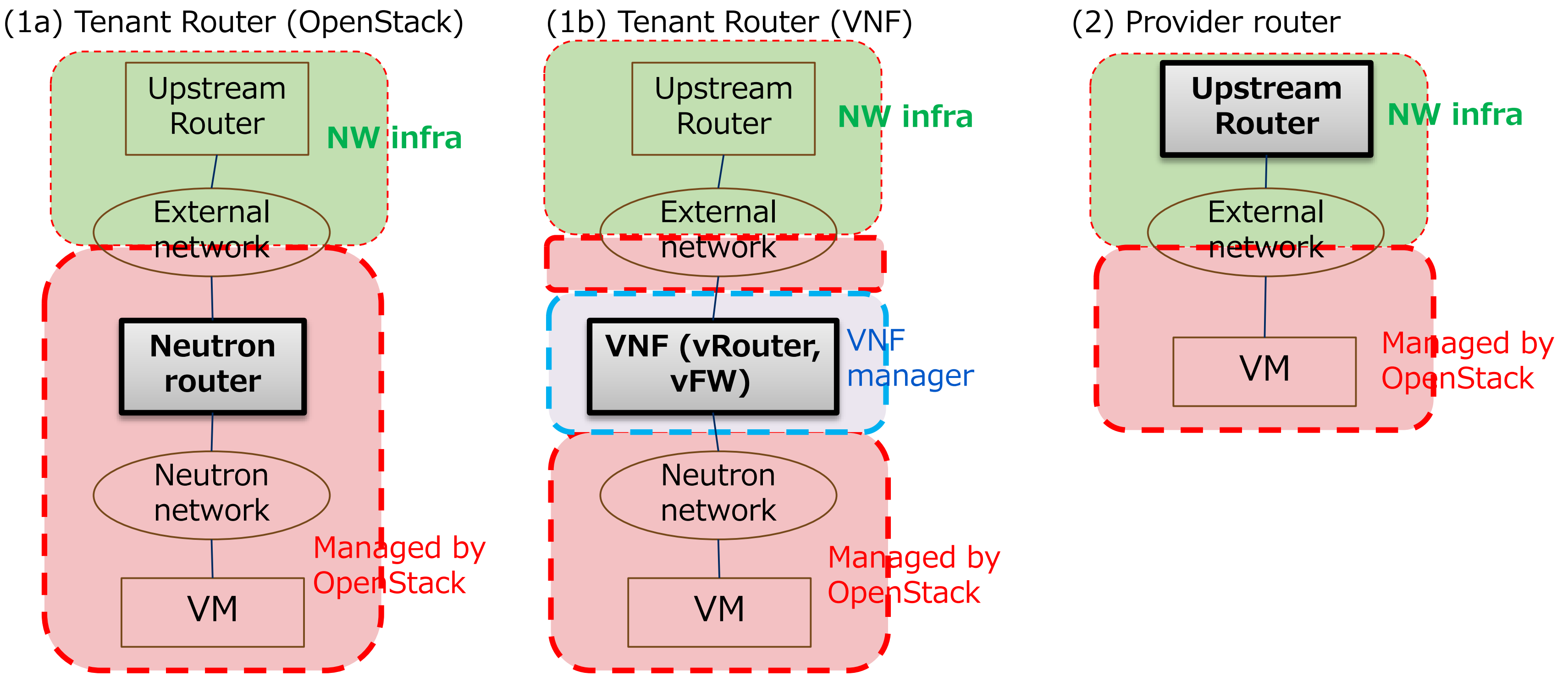
## (1a) Tenant Router (OpenStack)

- Neutron router; VNF integrated with neutron

## (1b) Tenant Router (VNF)

- Virtual appliance; not integrated with neutron

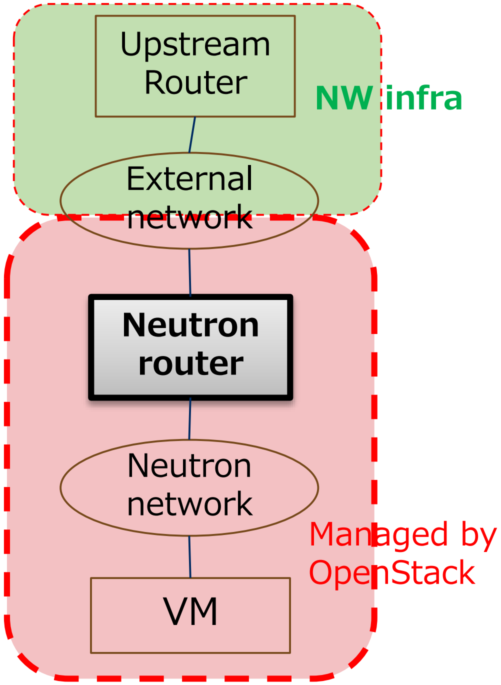
## (2) Provider Router



# IPv6 two modes: tenant neutron router

- Simplest case!
- Tenant router is implemented as neutron router
  - L3 agent (reference implementation)
  - Other 3rd party L3 plugin (if you use)
- Neutron provides both RA and DHCPv6
  - Ref implementation: RA = radvd, DCHPv6 = dnsmasq
- If 3rd party L3 plugin is provided, the behavior needs to be checked

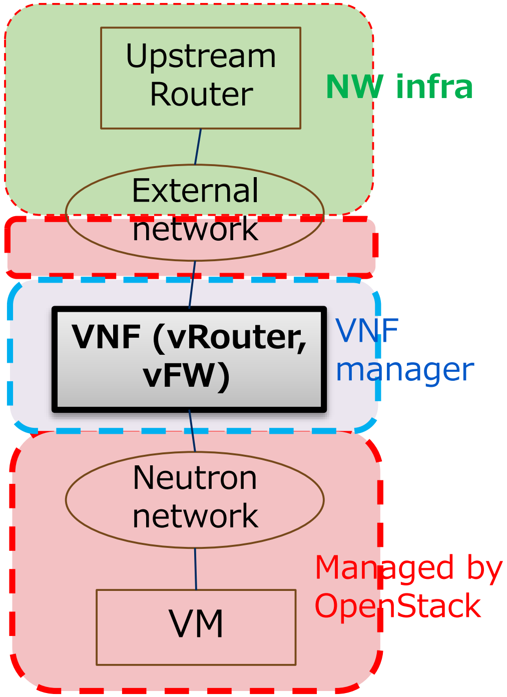
| ipv6_address_mode | ipv6_ra_ mode | Description (L3 agent) |
|---|---|---|
| SLAAC | SLAAC | Radvd runs on netns and provides RA |
| DHCPv6 stateless | DHCPv6 stateless | Radvd on router netns and dnsmasq on dhcp netns |
| DHCPv6 stateful | DHCPv6 stateful | Same as above |



# IPv6 two modes: tenant VNF router

- Tenant router is implemented as virtual appliance
- BUT it is not well integrated with neutron (= no L3 plugin)
- SLLAC works well
- DHCPv6-stateless / stateful : neutron DHCP (dnsmasq) serves too
  - If VNF provides DHCPv6, two DHCP servers works as a result and they may conflicts...

| ipv6_address _mode | ipv6_ra_ mode | Description |
|---|---|---|
| SLAAC | Off (N/S) | VNF router sends RA |
| DHCPv6 stateless | DHCPv6 stateless<br>Off | VNF router sends RA<br>neutron dnsmasq is configured<br>If VNF provides DHCPv6, the feature potentially conflicts··· |
| DHCPv6 stateful | DHCPv6 stateful<br>Off | VNF router sends RA<br>neutron dnsmasq is configured<br>If VNF provides DHCPv6, the feature potentially conflict··· |

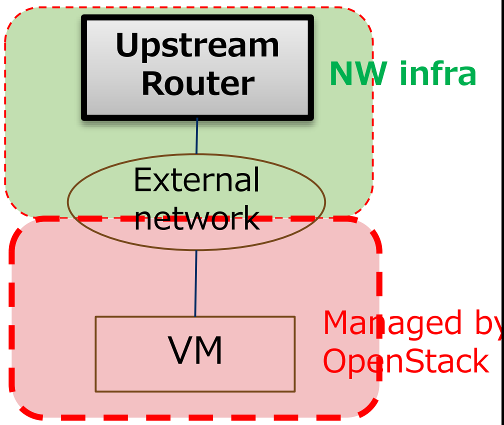


# IPv6 two modes: provider router

- Router on a provider network sends RA.
- From the perspective of neutron, same as “tenant VNF router” case
- SLLAC works well
- DHCPv6 stateless : DHCPv6 by neutron and upstream router may potentially conflict
- DHCPv6 stateful : neutron DHCPv6 is the only option?
  - There is no way to share address information between provider router and neutron

| ipv6_address_mode | ipv6_ra_mode | Description |
|---|---|---|
| SLAAC | Off (N/S) | provider router sends RA |
| DHCPv6 stateless | DHCPv6 stateless | provider router sends RA<br>neutron dnsmasq is configured<br>*If provider router provides DHCPv6, the feature potentially conflicts…* |
|  | Off |  |
| DHCPv6 stateful | DHCPv6 stateful | provider router sends RA<br>neutron dnsmasq is configured<br>If VNF provides DHCPv6, address from both will conflicts XXX |
|  | Off |  |

(2) Provider router



# In our case..

| VNF | SLAAC | DHCPv6-stateless |
|---|---|---|
| Neutron router | OK | OK |
| Cisco CSR1000V | OK (+DNS) | OK (+DNS) |
| NEC Intersec VM/SG | OK | - |
| Paloalto VM-100 | OK | - |

## (SLAAC)

- Not many VNF supports DNS option in RA message

## (DHCPv6-stateless)

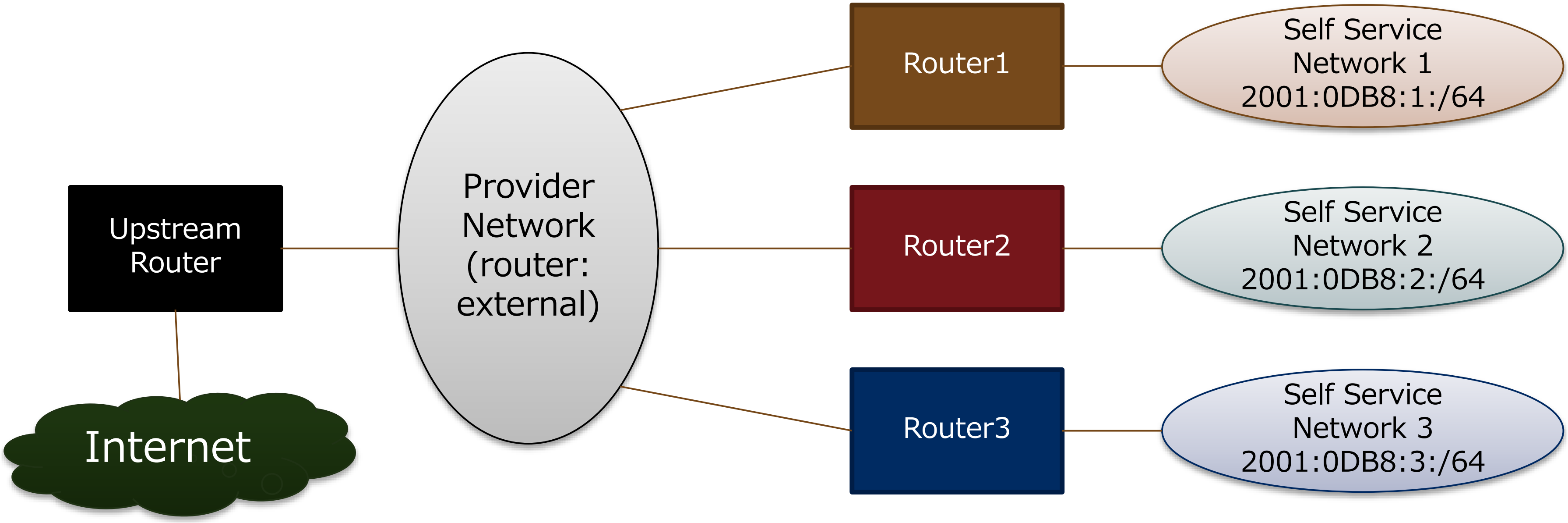
- Neutron DHCPv6-stateless does not provide DNS.. Needs investigation
- Some VNFs support SLAAC only. Cannot configure RA flags.

# Routing Considerations [1]

How to reach tenants’ network from the Internet?

# Dynamic routing to tenant IPv6 subnets

- How to reach tenants’ network?
- To deliver packets from the Internet to tenants’ networks, the upstream router must know routes to tenants’ networks
- In IPv6, tenant networks get GUA (global unique address) dynamically
- Route tables on the upstream router needs to be updated
- We need some mechanism to update routes automatically


Router1
Self Service Network 1 2001:0DB8:1:/64
Provider Network (router: external)
Upstream Router
Router2
Self Service Network 2 2001:0DB8:2:/64
Router3
Self Service Network 3 2001:0DB8:3:/64
Internet

# Dynamic routing to tenant IPv6 subnets

## Dynamic Routing

- BGP
- OSPF

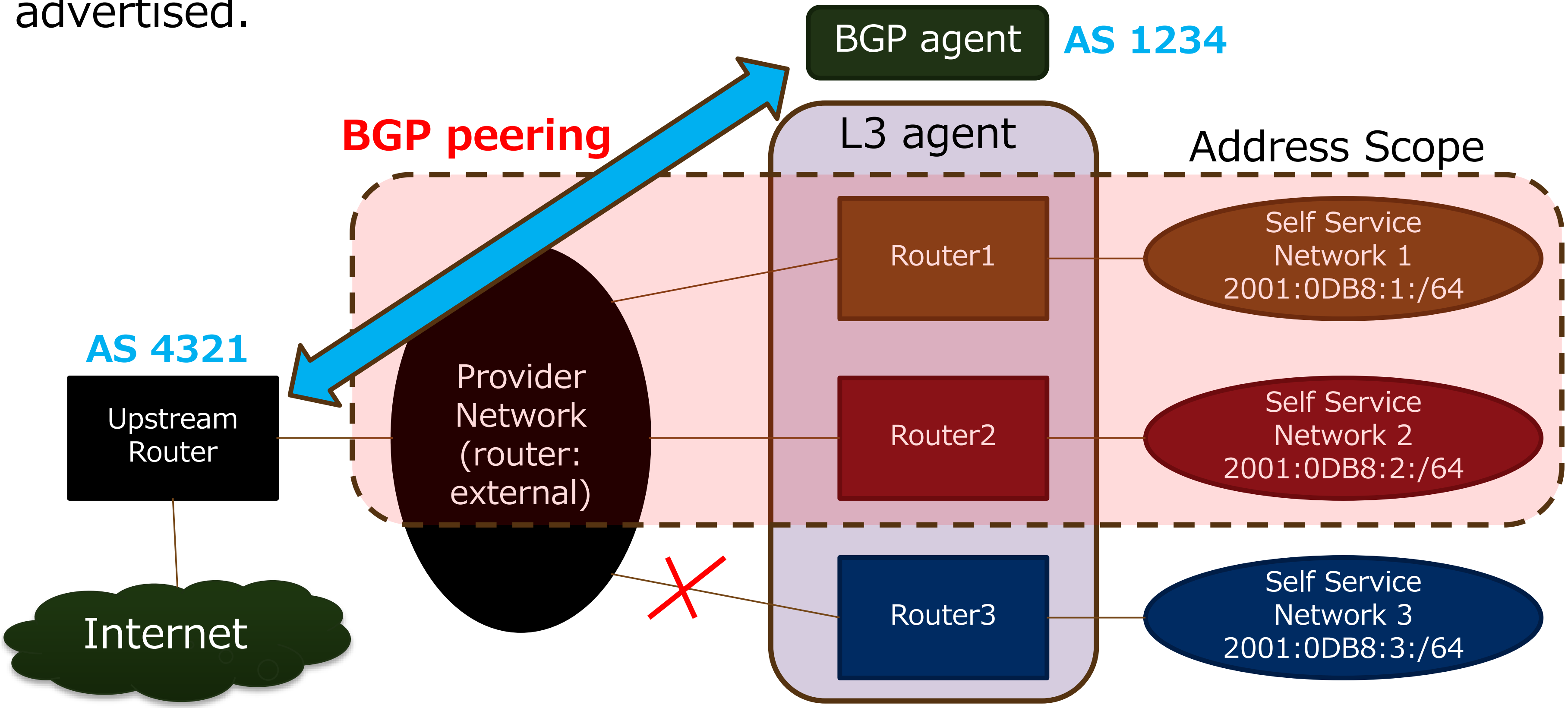
## IPv6 Prefix Delegation

## ~~Static Route (with some automation mechanism)~~

- Need some mechanism to monitor neutron-side changes

# Dynamic routing : BGP (neutron integration)

- Neutron provides BGP dynamic route advertisement
- Create a BGP peering between BGP agent (dr-agent) and upstream Router
- BGP agent advertises routes to tenant networks dynamically
- Only networks whose address scope is same as that of the external network are advertised. Networks on different address scope are not advertised.

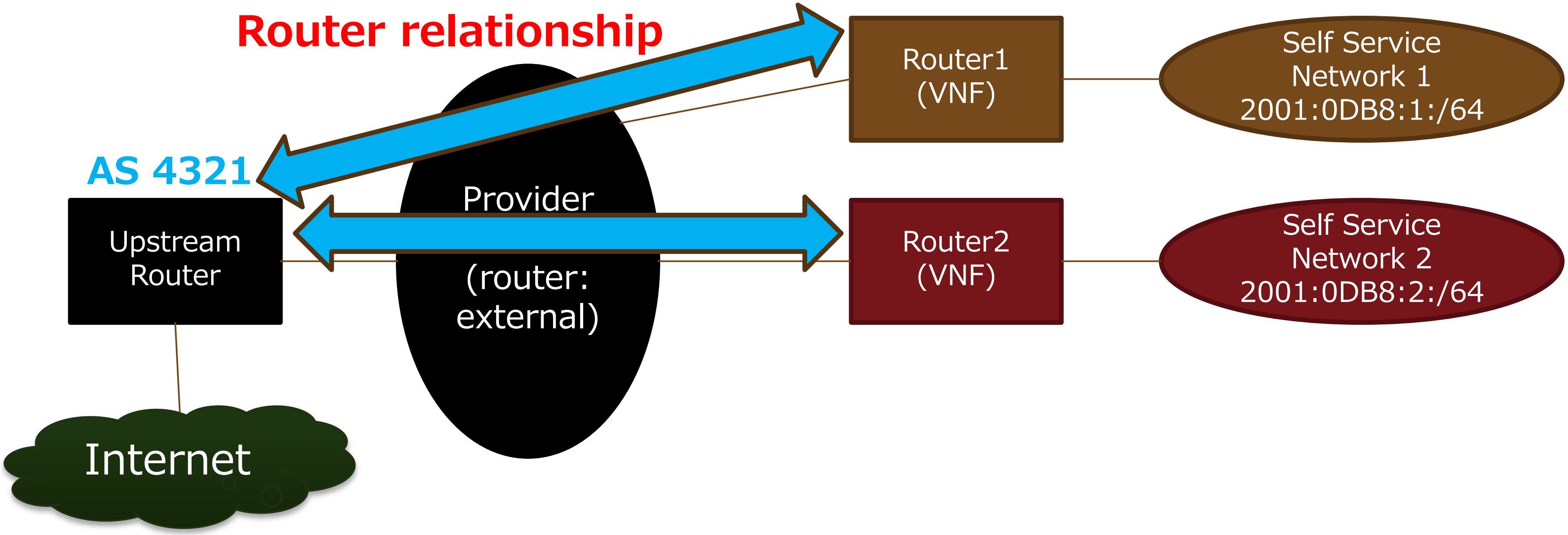


BGP agent
AS 1234
BGP peering
L3 agent
Address Scope
AS 4321
Upstream Router
Provider Network (router: external)
Router1
Router2
Router3
Self Service Network 1 2001:0DB8:1:/64
Self Service Network 2 2001:0DB8:2:/64
Self Service Network 3 2001:0DB8:3:/64
Internet

http://docs.openstack.org/newton/networking-guide/config-bgp-dynamic-routing.html

# Dynamic routing : OSPF

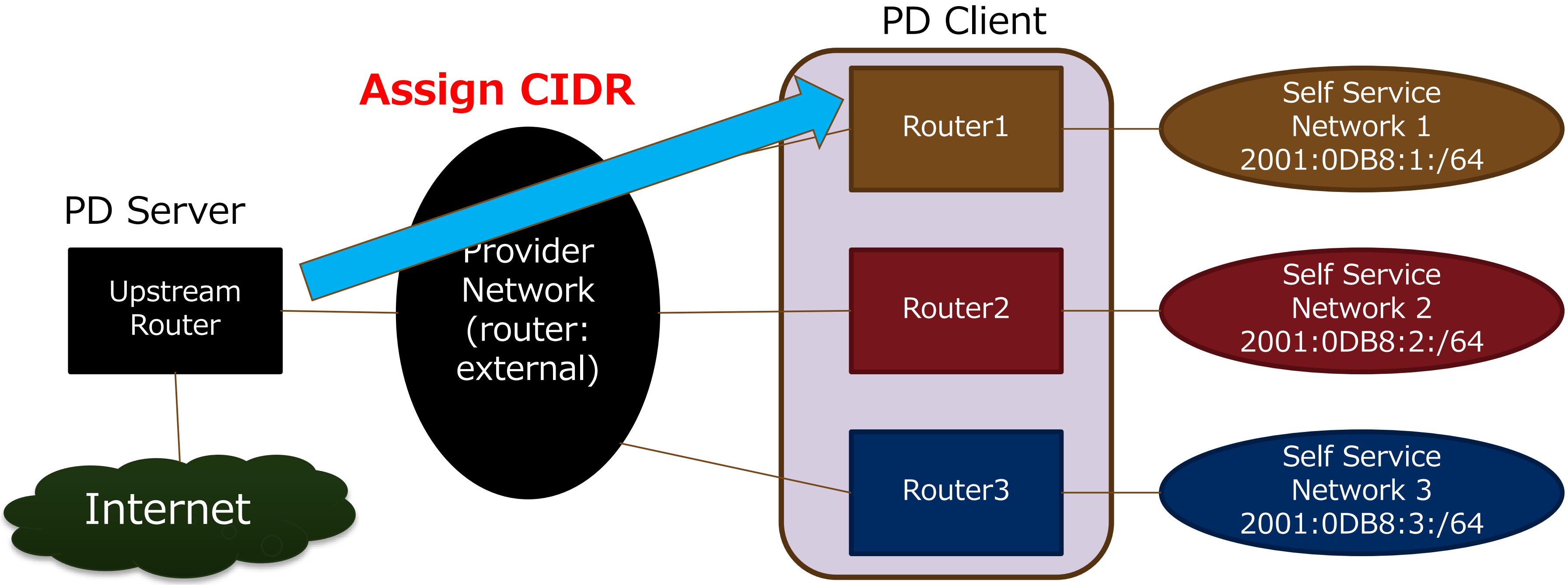
- OSPF can be used for dynamic routing
- Create router relationships between tenant and upstream routers when a tenant router is created
- Once a relationship is established, a route to a tenant network is configured to the upstream router
- Useful for small network where BGP is not preferred
- Most VNF router supports OSPF
- No neutron integration



http://docs.openstack.org/newton/networking-guide/config-bgp-dynamic-routing.html

# IPv6 Prefix Delegation (PD)

- Upstream router is a PD (prefix delegation) server and this manages IPv6 address ranges to be assigned to OpenStack tenant networks
- PD server assigns CIDR (normally /64 prefix) to PD client
- Neutron router acts as a PD client
- Upstream router sets up a route to PD client when assigning a prefix
  - The upstream router knows an external IP address of Neutron router (PD client)
  - LLA (Link local address) can be used as IP address of PD client
- Neutron integration


PD Client
Assign CIDR
PD Server
Upstream Router
Provider Network (router: external)
Router1
Router2
Router3
Self Service Network 1 2001:0DB8:1:/64
Self Service Network 2 2001:0DB8:2:/64
Self Service Network 3 2001:0DB8:3:/64
Internet

http://docs.openstack.org/newton/networking-guide/config-ipv6.html#prefix-delegation

# Comparison

## Dynamic routing (BGP)

- Neutron integration
- Depending on network policy of upstream network
- Some operators does not use BGP inside their network
- One AS is required for OpenStack deployment

## Dynamic routing (OSPF)

- No neutron integration
- OSPF is used for smaller deployment
- Most VNF router supports OSPF

## Prefix delegation

- Neutron integration
- Only simple topology is supported.
- Cannot handle nested tenant router
- Not a small number of VNF does not support prefix delegation

Choices depend on network policy and router types to be used

# Routing Considerations [2]

How to prevent unauthorized global addresses?

# How to prevent unauthorized global addresses?

- Neutron allows tenants to assign arbitrary IP addresses.

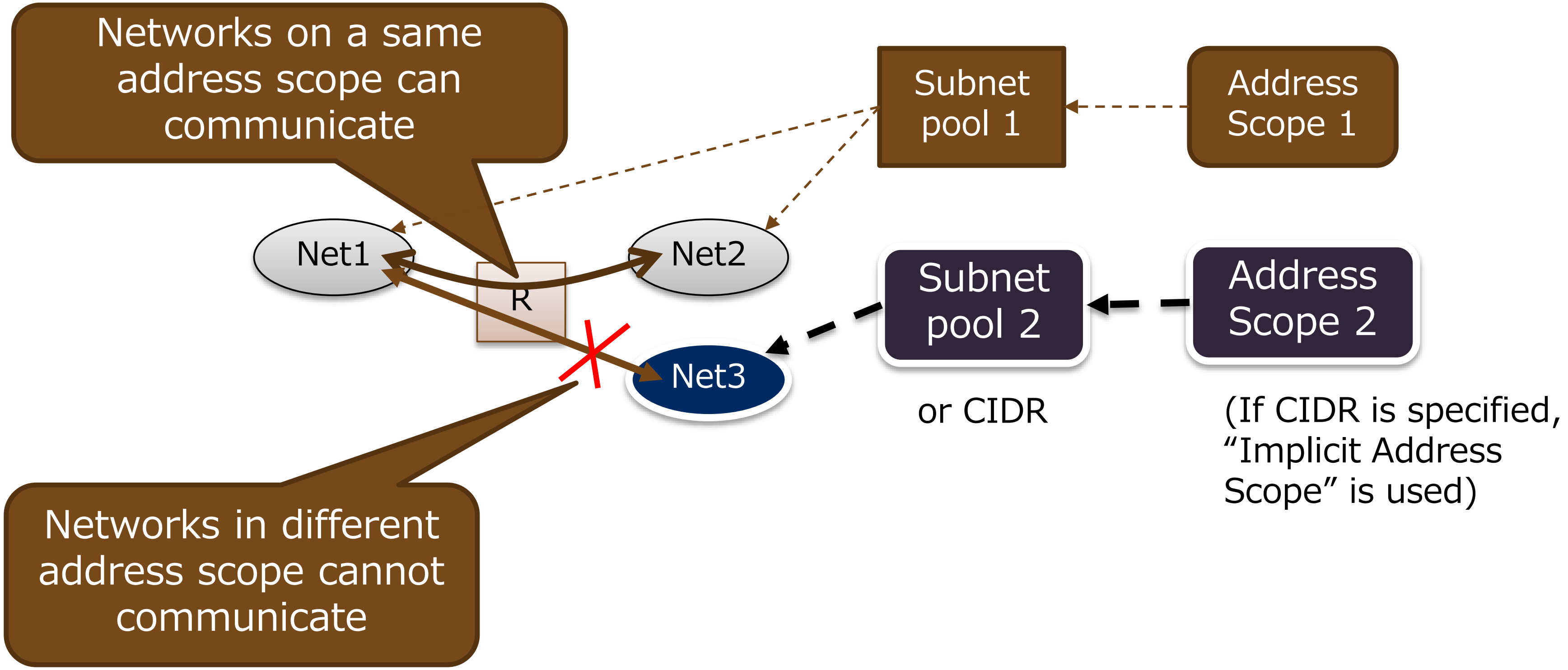
- In IPv6, global unique addresses are assigned to tenants from the predefined ranges.

- If some tenant assigns overlapping address ranges with other tenant, what happens?
- There needs a way to block **unauthorized** global unique address assigned by tenants in their own way.

# Neutron address scope

## Address Scope

- Concept to define which IP addresses can directly communicate each other
- Subnet pool is associated to some address Scope
- Router allows traffic among a same address scope
  - Router identifies an address scope of each router interface



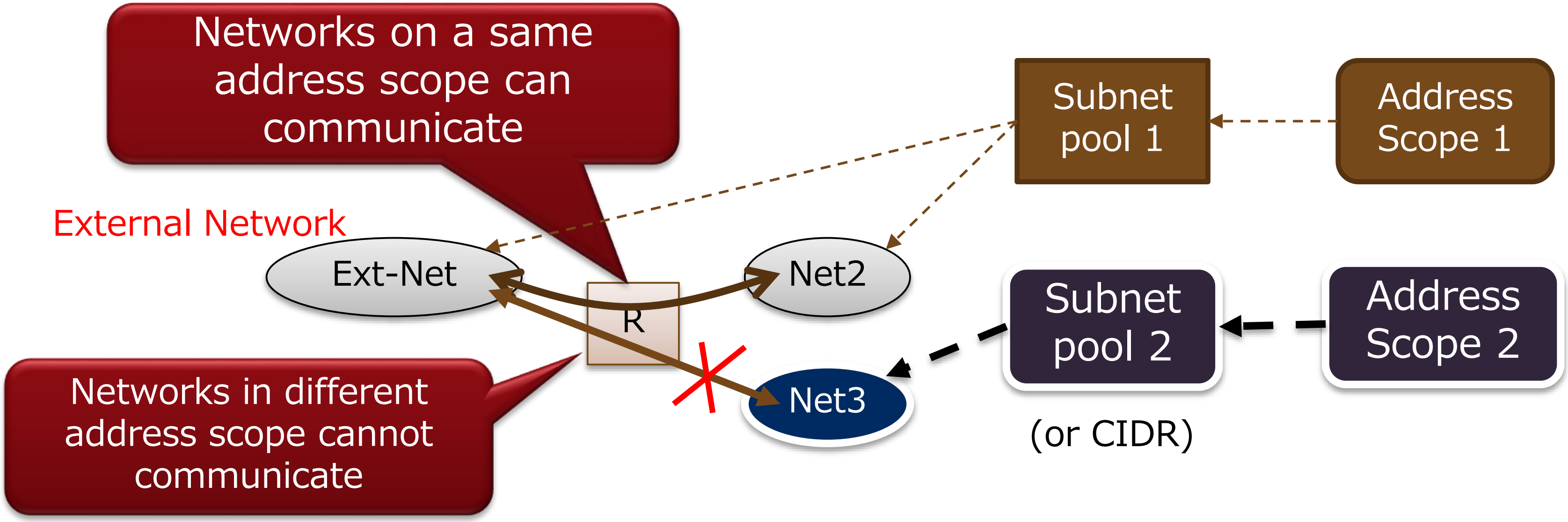
Networks on a same address scope can communicate

Networks in different address scope cannot communicate

(If CIDR is specified, “Implicit Address Scope” is used)

# How to prevent unauthorized global addresses?

- By using address scope, administrators can only allow traffic from IP address ranges they authorize.
- Administrator
  - Create a address scope
  - Create a subnet pool for external communication
  - Associate the subnet pool with the above address scope
  - Create a subnet of external network from the above subnet pool
- Tenant user
  - Create a subnet from the shared subnet pool (when communicating externally)
  - Subnet whose CIDR is specified explicitly cannot communicate with the Internet even if it has the same CIDR.


Networks on a same address scope can communicate
External Network
Ext-Net
R
Net2
Net3
Subnet pool 1
Address Scope 1
Subnet pool 2
Address Scope 2
(or CIDR)
Networks in different address scope cannot communicate

# How to prevent unauthorized global addresses?

## Create a address scope

- openstack address scope create --share --ip-version 6 address-scope-ip6

## Creaet a subnet pool for external communication

## Associate the subnet pool with the above address scope

- openstack subnet pool create --address-scope address-scope-ip6 --share --pool-prefix 2001:db8:a583::/48 --default-prefix-length 64 subnet-pool-ip6

## Create a subnet of external network from the above subnet pool

- openstack subnet create --subnet-pool subnet-pool-ip6 --ip-version 6 --disable-dhcp --network external-network external-subnet-ip6

## Create a subnet from the shared subnet pool

- openstack subnet create --subnet-pool subnet-pool-ip6 --ip-version 6 --ipv6-address-mode slaac --ipv6-ra-mode slaac --network my-network my-subnet-ip6

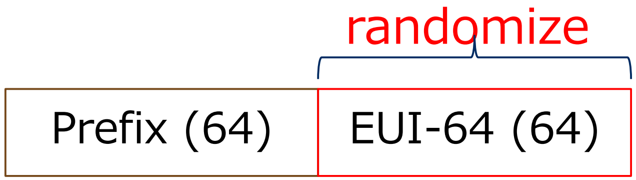
# Additional topics

# Additional topics

- For SLAAC/DHCPv6 stateless, /64 is actually the only unit assigned to tenant
  ➔ The required number of /64 prefix will be the number of tenants

- All IPv6 modes cannot be used necessarily, but neutron exposes all
  ➔ Some mechanism to expose available IPv6 mode to users

- Windows supports address randomization for EUI-64 address
  - It randomizes the lower 64 bits of addresses
  - Ephemeral address is also supported
  - It works ..
  - but a generated IPv6 address will be different from Neutron port information



- Most VNF does not provide Neutron L3 plugin
  ➔ Need a way to retrieve information from neutron
  - IPv6 RA flag (Managed, Other)
  - Address scope

# Summary

- This presentation share our experiences on IPv6
- Hope it helps introductions of IPv6 deployment

- Most things work well.
- There are several things remaining to be improved

- Let’s upstream it!
  - Feel free to file a bug to neutron!
- Let’s share our knowledge!
  - OpenStack Networking Guide is a good place

  - Latest release : https://docs.openstack.org/ocata/networking-guide/
  - Development version : https://docs.openstack.org/draft/networking-guide/

\Orchestrating a brighter world

NEC