Networking QoS, Liberty, Mitaka and Newton

Livnat Peer,
Senior Engineering Manager,
Red Hat

April, 2016
Network QoS
Network QoS
QoS

- No industry standard
  - Multiple ways to express bandwidth guarantees
    - OVS - min, max
    - Linux tc - rate, crate, burst, cburst

- Use Cases
  - Control the network resources
  - Tune the network to specific application type
  - Provide different SLAs
The Noisy Neighbor Problem

Nope. Both Mine
QoS in Neutron - Liberty

- Adding generic infrastructure that would be extensible for additional use cases
- Scope
  - Traffic within the hypervisor
  - Only traffic that leaves the VM (VM-egress)
  - No integration with Nova scheduler
A policy is a collection of rules that can be applied on a neutron port.

Policy attributes: Id, Name, Description, Shared, Tenant-Id

Policy can be associated with Neutron port or network.

```
# neutron qos-policy-create 'platinum'
    --description 'platinum QoS - charge a lot of $$'
```

```
# neutron port-update <port id> --qos-policy 'platinum'
# neutron net-update <net name> --qos-policy 'platinum'
```
QoS API & Data Model - Rules

- Rule is the building block of a policy
- Abstract QoS Rule
- QoS Bandwidth Limit Rule
  - max-kbps
  - max-burst-kb

```
# neutron qos-bandwidth-limit-rule-create <policy name> \
  --max-kbps 3000 \
  --max-burst-kbps 300
```
Data Model - Liberty - Summary

QoS Policy

Bandwidth Limit Rule

Network

Port
Rule Types

- Bandwidth Limit Rule
  - Liberty: OVS (ovs-vsctl) SR-IOV (ip link), Mitaka: Linux Bridge (tc)

```
# neutron qos-bandwidth-limit-rule-create <policy name> \
  --max-kbps 3000 \ 
  --max-burst-kbps 300
```
Rule Types

- Bandwidth Limit Rule
  - Liberty: OVS (ovs-vsctl) SR-IOV (ip link), Mitaka: Linux Bridge (tc)

```bash
# neutron qos-bandwidth-limit-rule-create <policy name> 
   --max-kbps 3000 
   --max-burst-kbps 300
```

- DSCP Marking Rule
  - value: even number between 0 and 56, except 2-6, 42, 44, and 50-54

```bash
# neutron qos-dscp-marking-limit-rule-create <policy name> 
  --dscp_mark <value>
```
Rule Types

- **Bandwidth Limit Rule**
  - Liberty: OVS (ovs-vsctl) SR-IOV (ip link), Mitaka: Linux Bridge (tc)
    - Example command:
      ```
      # neutron qos-bandwidth-limit-rule-create <policy name> \
      --max-kbps 3000 \
      --max-burst-kbps 300
      ```

- **DSCP Marking Rule**
  - Value: even number between 0 and 56, except 2-6, 42, 44, and 50-54
    - Example command:
      ```
      # neutron qos-dscp-marking-limit-rule-create <policy name> \
      --dscp_mark <value>
      ```

- **There is an option to see all supported Rule types**
  - Example command:
    ```
    # Neutron qos-available-rule-types
    ```
Data Model

Diagram:
- QoS Policy
  - Network (n)
  - Port (n)
- Bandwidth Limit Rule
- DSCP Marking Rule

Relationships:
- QoS Policy to Network: 1
- QoS Policy to Port: 1
- Bandwidth Limit Rule to QoS Policy: 1
- DSCP Marking Rule to QoS Policy: 1
Policy propagation

- Changes to the Policy immediately propagate to the ports
Policy propagation

- Changes to the Policy immediately propagate to the ports

Attach a policy to a port
Policy propagation

Update Policy
Role Based Access Control (RBAC)

- **QoS permission model (Liberty)**
  - By default only cloud admin can create a QoS policy
  - Shared QoS policy
  - The default behaviour managed in policy.json file

- **RBAC mechanism - introduced in Liberty**
  - A policy framework that allows both operators and users to grant access to resources for specific projects
  - The generic mechanism was added in Liberty with ‘Network’ as the first resource supporting this mechanism
QoS Policy - RBAC Support

- Support was added in Mitaka

```bash
# neutron qos-policy-create secret_policy
```

```bash
# neutron rbac-create --target-tenant <tenant-uuid> \
    --action access_as_shared \
    --type qos-policy \
    <qos-policy-uuid>
```
Future - Min Bandwidth guarantee

- Seems like a simple change

```
# neutron qos-bandwidth-limit-rule-create <policy name> \
  --max-kbps 3000 --max-burst-kbps 300 \n  --min-kbps 1000
```

- Challenges
  - Support ‘borrowing’
  - Requires influencing VM placements
Min Bandwidth guarantee

- Seems like a simple change

```bash
# neutron qos-bandwidth-limit-rule-create <policy name> \
--max-kbps 3000 --max-burst-kbps 300 \n--min-kbps 1000
```

- Challenges
  - Support ‘borrowing’
  - Requires influencing VM placements

![Diagram showing comp1 and comp2 with 10G each]
Min Bandwidth guarantee

- Seems like a simple change

  ```bash
  # neutron qos-bandwidth-limit-rule-create <policy name> \ 
  --max-kbps 3000 --max-burst-kbps 300 \ 
  --min-kbps 1000
  ```

- Challenges
  - Support ‘borrowing’
  - Requires influencing VM placements

```
# neutron qos-bandwidth-limit-rule-create <policy name> \ 
   --max-kbps 3000 --max-burst-kbps 300 \ 
   --min-kbps 1000
```

comp1

```
min=7G
```

comp2

```
10G each
```
Min Bandwidth guarantee

- Seems like a simple change

```
# neutron qos-bandwidth-limit-rule-create <policy name> \
--max-kbps 3000 --max-burst-kbps 300 \n--min-kbps 1000
```

- Challenges
  - Support ‘borrowing’
  - Requires influencing VM placements
Min Bandwidth guarantee

- Seems like a simple change

```bash
# neutron qos-bandwidth-limit-rule-create <policy name> \
    --max-kbps 3000 --max-burst-kbps 300 \n    --min-kbps 1000
```

- Challenges
  - Support ‘borrowing’
  - Requires influencing VM placements

Diagram:
- VM 1
  - min=7G

- comp1
  - 10G each
- comp2
  - 10G each
Min Bandwidth guarantee

- Seems like a simple change

```
# neutron qos-bandwidth-limit-rule-create <policy name> \
   --max-kbps 3000 --max-burst-kbps 300 \n   --min-kbps 1000
```

- Challenges
  - Support ‘borrowing’
  - Requires influencing VM placements

Diagram:

- VM 1: min=7G
- VM 2: min=5G
- comp1: 10G each
- comp2: 10G each
Min Bandwidth guarantee

- Seems like a simple change

  ```
  # neutron qos-bandwidth-limit-rule-create <policy name> \
  --max-kbps 3000 --max-burst-kbps 300 \ 
  --min-kbps 1000
  ```

- Challenges
  - Support ‘borrowing’
  - Requires influencing VM placements

VM 1
- min=7G
- comp1

VM 2
- min=5G
- comp2

10G each
Min Bandwidth guarantee

- Seems like a simple change

```bash
# neutron qos-bandwidth-limit-rule-create <policy name> \
  --max-kbps 3000 --max-burst-kbps 300 \n  --min-kbps 1000
```

- Challenges
  - Support ‘borrowing’
  - Requires influencing VM placements

---

**Diagram:**

- **comp1**
  - VM 1
  - min=7G
- **comp2**
  - VM 2
  - min=5G

Overall, each 10G
Min Bandwidth guarantee

- Nova scheduler
  - Resource Provider Framework =>

```
resource-classes
resource-providers
compute-node-inventory
recource-providers-allocations
```
Current/Future Work

- Traffic classifiers
  - https://review.openstack.org/#/c/238812
  - https://review.openstack.org/#/c/190463/
- Shaping incoming traffic
- QoS VLAN 802.1p Support
- Many other ideas and requests - https://bugs.launchpad.net/neutron/+bugs?field.tag=qos
Thank You
Resources

- **Neutron QoS API Extension** - Neutron spec
- **Ajo’s Blog** - Neutron Quality of Service coding sprint
- **DSCP Marking** - Neutron spec
- **Add Classifier Resource** - Neutron spec
- **User Guide for QoS**
- **The noisy neighbor problem**
- **RBAC policy** - OpenStack Manual Documentation
- **Nova resource Provider Framework**
Resources

- Nova Resource Providers Framework
  - [Link](https://review.openstack.org/#/c/253187/14)
  - [Link](https://review.openstack.org/#/c/271779/4/specs/newton/approved/resource-providers-allocations.rst)
  - [Link](https://review.openstack.org/#/c/225546/10/specs/mitaka/approved/resource-providers.rst)
  - [Link](https://review.openstack.org/#/q/topic:bp/generic-resource-pools)