QoS support for Open vSwitch OpFlex

New and Changed Information 2

About QoS support for Open vSwitch OpFlex 2

Benefits of using QoS support for Open vSwitch OpFlex 2

QoS support for Open vSwitch OpFlex Limitations and Restrictions 2

Prerequisites for Configuring QoS support for Open vSwitch OpFlex 2

QoS support for Open vSwitch OpFlex Configuration Workflow 3

Configuring the QoS support for Open vSwitch OpFlex in OpenStack 3

Configuring the QoS support for Open vSwitch OpFlex in Kubernetes 4

Verifying the QoS support for Open vSwitch OpFlex in OpenStack 5

Verifying the QoS support for Open vSwitch OpFlex in Kubernetes 6

Revised: June 24, 2021

New and Changed Information

The following table provides an overview of the significant changes to this guide up to this current release. The table does not provide an exhaustive list of all changes that are made to the guide or of the new features up to this release.

Table 1: New Features and Changed Behavior

Openstack Unified Plug-in Version	Cisco ACI CNI Plug-in Version	Feature	Description
5.1(3)	5.1(3)	QoS support for Open vSwitch OpFlex	This guide became available.

About QoS support for Open vSwitch OpFlex

In OpenStack platform (OSP) and Kubernetes platform (K8S), the QoS policy can be created with QoS rule types such as 'Bandwidth Limit', 'Minimum bandwidth', 'DSCP marking' and 'Max burst'. The QoS policy can be applied directly to an endpoint (EP) or an endpoint group (EPG) seamlessly using Cisco APIC.

Benefits of using QoS support for Open vSwitch OpFlex

Using QoS support for Open vSwitch OpFlex provides several benefits:

- Limiting bandwidth of a chatty endpoint (noisy neighbor type issues where one of the endpoint is consuming majority of the bandwidth and causing low bandwidth for other endpoints).
- Use the dscp marker to affect the priority of the packet so that the best effort of delivery and low latency delivery is possible.

QoS support for Open vSwitch OpFlex Limitations and Restrictions

Be aware of the following issues when configuring QoS support for Open vSwitch OpFlex:

- QoS is not supported on any Layer 3 ports. This includes ports used for router interfaces and gateways, as well as floating IPs.
- Limitations of OVS for ingress policing is applicable (egress for endpoint).

  For more information, see the Ingress Policing section in the latest `ovs-vswitchd.conf.db` document by *Open vSwitch*.
- QoS is not supported for ports that belong to an external network in OpenStack.
- Configuration of Min bandwidth parameter is not supported.

Prerequisites for Configuring QoS support for Open vSwitch OpFlex

You must complete the following tasks before you configure QoS support for Open vSwitch OpFlex:

- For OpenStack, Unified Plug-in version 5.1.3.
- For Kubernetes, Cisco ACI CNI Plug-in version 5.1.3.
- OpenStack and K8s (CNI) components should be correcponding to Cisco APIC 5.1(3).

  For more information, see the Cisco ACI Virtualization Matrix:

  https://www.cisco.com/c/dam/en/us/td/docs/Website/datacenter/aci/virtualization/matrix/virtmatrix.html

QoS support for Open vSwitch OpFlex Configuration Workflow

There are two ways to configure QoS:

- Endpoint level
- EPG level

1. Create a QoS policy.
2. Create egress, ingress, dscp rules, and associate it with QoS policy.

   In OpenStack, to apply the QoS policy at the endpoint level, the policy is assigned to the port corresponding to the endpoint. To apply QoS policy at the EPG level, the policy is applied to the network. When applied to a network the QoS policy would be applied to all the endpoints in that network.

   If both are applied, then the endpoint level QoS is prioritized.

   For more information, see Configuring the QoS support for Open vSwitch OpFlex in OpenStack, on page 3.

   In Kubernetes, QoS policy can be applied at the pod level or at the namespace level. If applied at namespace level, policy will be applied to all the pods in that namespace. Filters can be used to selectively apply a policy on the pods in a namespace. Annotations can be used to apply a policy at the pod level.

   For more information, see Configuring the QoS support for Open vSwitch OpFlex in Kubernetes, on page 4.
3. To verify the QoS support for Open vSwitch OpFlex, see Verifying the QoS support for Open vSwitch OpFlex in OpenStack, on page 5 or Verifying the QoS support for Open vSwitch OpFlex in Kubernetes, on page 6.

Configuring the QoS support for Open vSwitch OpFlex in OpenStack

This section describes how to configure the QoS support for Open vSwitch OpFlex in OpenStack.

Procedure

Step 1 QoS is configured from OpenStack CLI, enter the following commands:

Example:

```
$ openstack network qos policy create bw-limiter
$ openstack network qos rule create --type bandwidth-limit --max-kbps 3000 \
--max-burst-kbits 2400 --egress bw-limiter
$ openstack network qos rule create --type dscp-marking --dscp-mark 26 bw-limiter
$ openstack network qos rule create --type bandwidth-limit --max-kbps \
4000 --max-burst-kbits 400 --ingress bw-limiter
```

Step 2 To attach it to a network and to apply QoS at the EPG level, enter the following command:

Example:

```
$ openstack network set --qos-policy bw-limiter private
```

Step 3 To attach it to a port and to apply QoS at the endpoint level, enter the following command:

Example:

```
$ openstack port set -qos-policy bw-limiter b36de9b3-79a1-473e-ba7c-0a4d1adfe81d
```

Configuring the QoS support for Open vSwitch OpFlex in Kubernetes

This section describes how to configure the QoS support for Open vSwitch OpFlex in Kubernetes.

Procedure

Step 1 Custom resource yaml file creates QoS policies and associates to pods in the namespace. The yaml file is applied using following command:

```
$ kubectl apply -f <yaml_file>
```

Step 2 Applying to all pods in a `namespace`. Apply the following custom resource yaml file:

```
apiVersion: aci.qos/v1
kind: QosPolicy
metadata:
    name: qos-policy7
    namespace: default
spec:
    ingress:
        policing_rate : 1000
        policing_burst: 1000
    egress:
        policing_rate:  1000
        policing_burst: 2000
    dscpmark: 8
```

Step 3 Applying to a single pod. This applies all the QoS policies ascociated with the `name` '`default`' to '`pod-a`:

```
apiVersion: v1
kind: Pod metadata:
name: pod-a
    annotations:
      "opflex.cisco.com/qospolicy": '{"policy-space":"defaultTenant","name":"default"}'
spec:
    ingress:
        policing_rate : 1000
        policing_burst: 1000
    egress:
        policing_rate: 1000
        policing_burst: 2000
    dscpmark: 8
```

Below is an example of where the QoS policies are applied to all pods in the `namespace` set to '`default`' and where '`role`' is set to '`db`':

Example:

```
apiVersion: aci.qos/v1
kind: QosPolicy
metadata:
   name: qos-policy7
   namespace: default
spec:
   podSelector:
      matchLabels:
      role: db
   ingress:
        policing_rate : 1000
        policing_burst: 1000
   egress:
        policing_rate:  1000
        policing_burst: 2000
   dscpmark: 8
```

Verifying the QoS support for Open vSwitch OpFlex in OpenStack

This section describes how to verify the QoS support for Open vSwitch OpFlex in OpenStack.

Verification of the QoS policy should be started from compute node hosting the instances to which the ports belong. The <*tap-interface-name*> is the tap interface created for the port. The endpoint file generated for an endpoint can be used to get the tap interface name under section `access-interface` for a port. The endpoint file name will have the port UUID in it. The location of endpoint files are in the `/var/lib/opflex-agent-ovs/endpoints/` directory inside the `ciscoaci_opflex_agent` <*verify-parameters-on-the-interface*> container.

Note The policing direction (ingress/egress) in the QosPolicy configuration is applied to OVS in the opposite direction, i.e. ingress in the QosPolicy, is egress in OVS. This is expected.

Procedure

Step 1 Engress policing rate and burst, enter the following command:

```
[root@overcloud-compute-0 ~]# ovs-vsctl list interface <tap-interface-name>
```

Step 2 Igress policing rate and burst, enter the following commands:

```
[root@overcloud-compute-0 ~]# ovs-vsctl list port <tap-interface-name>
[root@overcloud-compute-0 ~]# ovs-vsctl list qos <qos-id>
[root@overcloud-compute-0 ~]# ovs-vsctl list queue <queue-id>
```

Step 3 Verify the dscp marking. Dump the flows for integration bridge and grep the openflow port number for the port and check `mod_nw_tos` action in the flow. Flow will have the mark ***4** value against `mod_nw_tos` field. Get the `openflow-port-number` for a port from '`ofport`' column in '`interface`' table for the port.

```
[root@overcloud-compute-0 ~]# ovs-ofctl dump-flows br-access -O OpenFlow13 | grep
in_port=<openflow-port-number>
```

Sample output:

```
cookie=0x0, duration=3162.620s, table=0, n_packets=0, n_bytes=0, priority=65535,ip,in_port=51
actions=set_field:16->ip_dscp,resubmit(51,1)
```

For the `dscp-marking configured = 26`, the flow will have 104. (26*4)

Verifying the QoS support for Open vSwitch OpFlex in Kubernetes

This section describes how to verify the QoS support for Open vSwitch OpFlex in Kubernetes.

Verification of the QoS policy should be started from compute node hosting the instances to which the ports belong. The <*tap-interface-name*> is the tap interface created for the port. The endpoint file generated for an endpoint can be used to get the tap interface name under section `access-interface` for a port. The endpoint file name will have the port UUID in it. The location of endpoint files are in the `/var/lib/opflex-agent-ovs/endpoints/` directory inside the `ciscoaci_opflex_agent` <*verify-parameters-on-the-interface*> container.

Note The policing direction (ingress/egress) in the QosPolicy configuration is applied to OVS in the opposite direction, i.e. ingress in the QosPolicy, is egress in OVS. This is expected.

In Kubernetes, OVS configurations are verified inside open-vswitch pod. OVS configuration verification commands are same as OpenStack as mentioned in the Verifying the QoS support for Open vSwitch OpFlex in OpenStack, on page 5. Follow the steps below to get a shell inside the open-vswitch pod:

Procedure

Step 1 Look up the pods, enter the following command:

```
$ kubectl get pods -A
```

Step 2 SSH into the open-vswitch pod, enter the following commands:

```
$ kubectl exec -it -n <namespace> <pod-name> /bin/sh
```

Example:

```
$ kubectl exec -it -n aci-containers-system aci-containers-openvswitch-l2lxk /bin/sh
```

Americas Headquarters
Cisco Systems, Inc.
San Jose, CA 95134-1706
USA

Asia Pacific Headquarters
CiscoSystems(USA)Pte.Ltd.
Singapore

Europe Headquarters
CiscoSystemsInternationalBV
Amsterdam,TheNetherlands

Cisco has more than 200 offices worldwide. Addresses, phone numbers, and fax numbers are listed on the Cisco Website at www.cisco.com/go/offices.