Focus Lab 17: QoS

Overview

In this lab, you will gain knowledge about zone-based and flow-based QoS configuration. QoS provides relative bandwidth guarantee to application transfers and controls latency experienced by fibre channel traffic. You will learn how to prioritize one application over another using QoS and FCC Congestion Control.

The following features are covered in this Focus Lab:

- FCC
- Flow based QoS
- Zone based QoS

ieMentor Storage VRACK support

Our VRACK is fully compliant with this lab.

Theory: QoS

We want you to memorize the following statement about QoS on Cisco MDS switches.

For MDS QoS to be effective, all competing traffic must enter the switch through the same ingress port X and exit through the same egress port Y.

While MDS QoS is very simple to configure it may be difficult to understand its operation. Keep in mind that FC traffic flows in two directions, from a host to a storage device and from a storage device to a host. MDS QoS will not work for all topologies and traffic flow scenarios due to the hardware architecture and Fibre Channel nature.

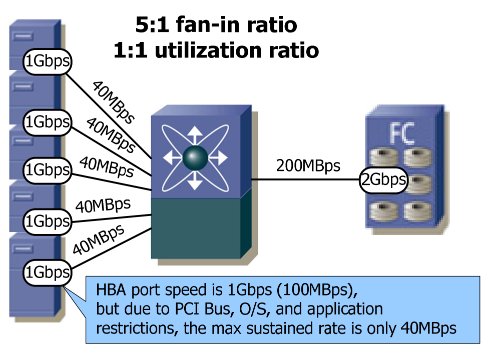
Don't just configure QoS and assume that it works. Analyze your topology and traffic flows, pinpoint congestion spots, and decide if QoS is actually going to help you.

Oversubscription:

- Host Oversubscription. Slow PCI bus in the server is typically the bottleneck and doesn't allow HBA to work at its full speed. Furthermore, if you have another PCI card (for example Gigabit Ethernet) on the same bus, the HBA performance will decline even more. Typically, an HBA with 2Gbps (200MBps) specification can transfer data at 20 to 40MBps max sustained rate.

- Fan-In ratio – number of hosts (initiator ports) to storage devices (target ports). Acceptable fan-in ratio depends on the application and host requirements. Typical fan-in ratios range from 6:1 to 12:1. For example:
    - 5 hosts, each capable of 40MBps sustained transfer rate
    - 1 storage port with a 2Gbps connection, capable of 200MBps transfer rate
    - Fan-in ratio is 5:1
    - Utilization ratio is 1:1 ([40MBps * 5] / 200MBps)
    - Fan-in ratio is acceptable

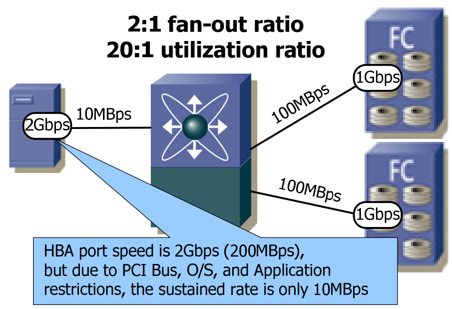
Figure 230: Fan-in ratio



- Fan-Out ratio – number of storage devices (target ports) to hosts (initiator ports). Acceptable fan-out ratio also depends on the application and host requirements. For example:
    - 1 host capable of 10MBps sustained transfer rate

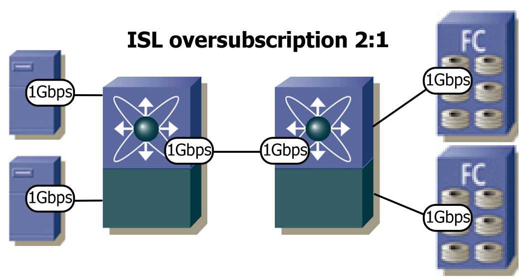
- 2 storage ports with 1Gbps connection each (100MBps + 100MBps transfer rate)
- Fan-out ratio is 2:1
- Utilization ratio is 20:1 ([100MBps + 100MBps] / 10MBps)
- Acceptability of this utilization ratio depends on the application requirements

Figure 231: Fan-out ratio



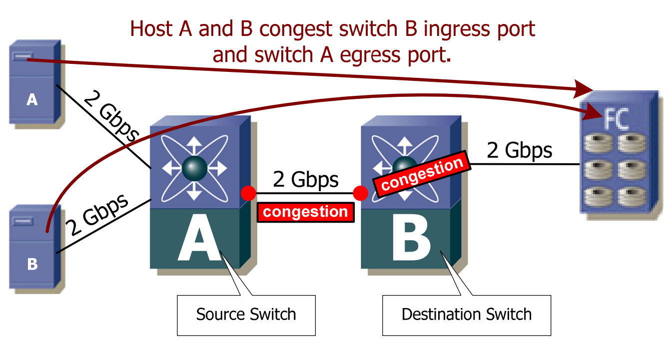
- ISL Oversubscription – number of devices sharing one ISL link. Typical ISL oversubscription ratios range from 8:1 to 16:1.

Figure 232: ISL oversubscription



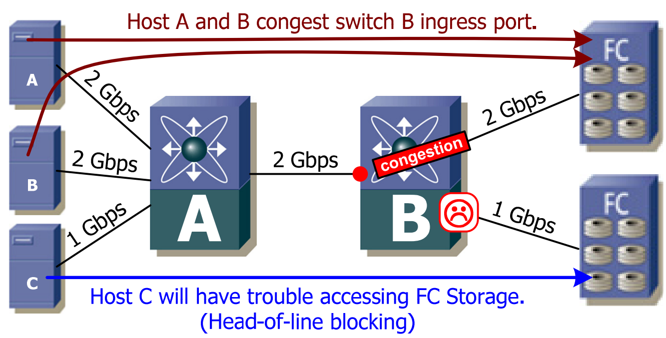
Congestion happens when multiple hosts on one ingress port are fighting for a storage device on one egress port, thus starving ingress port bandwidth and blocking other legitimate traffic from reaching other egress ports. It can also happen when multiple storage devices on a single ingress port are fighting for a host on an egress port, thus starving ingress port bandwidth and blocking other legitimate traffic from reaching other egress ports.

Figure 233: Congestion



Congestion can lead to Head-of-line (HOL) blocking. HOL blocking occurs when congestion on one link negatively impacts the throughput on a different link.

Figure 234: Head-of-line blocking



What does the statement "multiple hosts on one ingress port" mean? It means that you have to have at least two switches with a trunk between them for congestion to occur. You will later see that QoS is almost useless in a single switch topology.

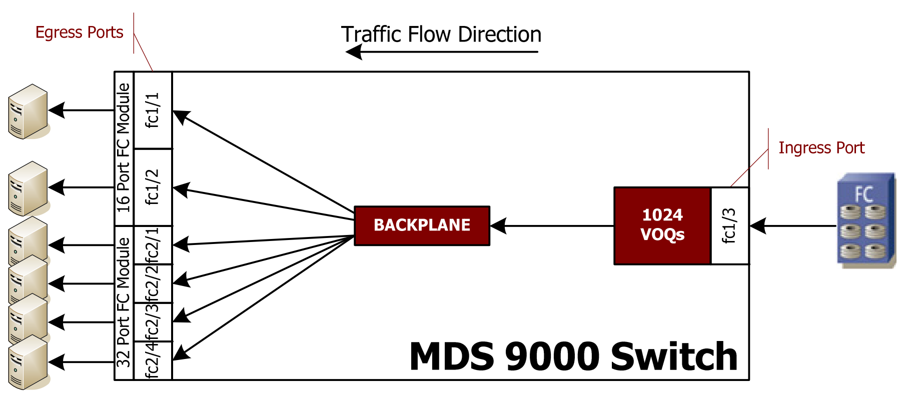
Cisco MDS switch backplane architecture was designed to provide maximum throughput. Congestion and HOL blocking will never occur on the MDS switch backplane. Congestion will only occur on the Fibre Channel interfaces. Some other vendor switches are susceptible to congestion and HOL blocking on the backplane due to hardware design limitations.

Cisco offers two proprietary features for traffic management: FCC and QoS.

- FCC (Fibre Channel Congestion Control). Intelligent process of throttling down traffic at its source if there is congestion somewhere in the path. FCC controls bandwidth. FCC is part of the free SAN-OS package.
- QoS (Quality of Service). Prioritizing data traffic (three data queues + one control queue) on each ingress port. QoS controls latency. ***QoS is inefficient without FCC***. QoS requires enterprise license.

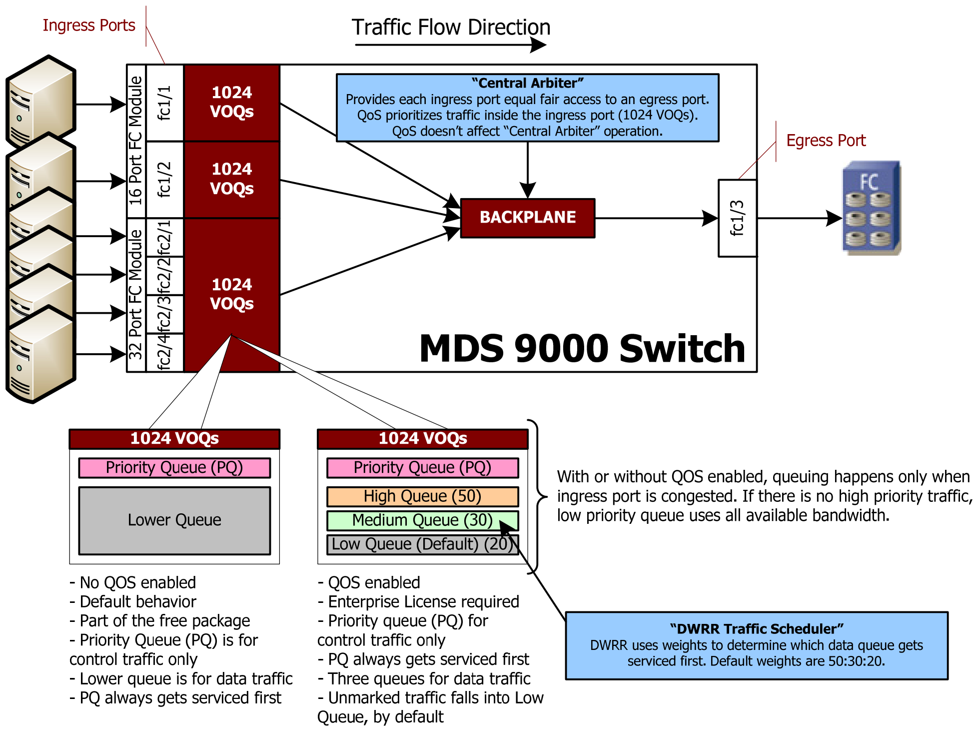
Cisco MDS Switches have 1024 VOQs (Virtual Output Queues) on each ingress port.

Figure 235: Virtual Output Queues on the MDS 9000 switches



Central Arbiter provides fair access from each ingress port, through the backplane, to the egress port. Central Arbiter is not controlled by QoS. Each ingress port has an **equal chance** to send frames to an egress port. Egress ports don't have queues. When its turn comes, ingress port VOQ sends the frame straight through the backplane to the egress port towards the destination. This architecture is drastically different from Ethernet switches that have ingress and egress hardware queues.

Figure 236: Virtual Output Queues on the MDS 9000 switches



DWRR (Deficit Weighted Round Robin) controls queuing inside each of the ingress port VOQs:

- By default, with QoS disabled, each ingress port has a Priority Queue and a Lower Queue. PQ is sometimes called Absolute queue, it can be used by control traffic only. PQ always gets serviced first. Control traffic is typically generated by the supervisors (Class F frames). This traffic is not intended for hosts or storage devices. Control traffic can also be generated from other vendor switches. The lower queue is used for the Data traffic. DWRR traffic scheduler's job is pretty simple in this default mode. It just has to make sure that PQ is always serviced first.

- With QoS enabled (Enterprise license required), Priority Queue is still intended only for the control traffic and always gets serviced first. But the Lower Queue gets split into three DWRR queues: High, Medium and Low. DWRR uses weights when deciding which of the three queues to service next. Default weights are 50:30:20 (High:Medium:Low). This means that the High Queue will be serviced 2.5 times more often than the Low queue. Default weight distribution can be changed. For example, you could make is 70:20:10. Keep in mind, that QoS kicks in only when there is a congestion on the given ingress port. If High queue is not being used, Low priority traffic will use all available bandwidth when sending traffic to the egress port. By default, all unmarked traffic goes into the Low priority queue.

"Congestion on the egress port" automatically leads to "congestion on the ingress port" because egress ports have no queues of their own. If an egress port is congested (all bandwidth is used), the ingress port trying to send frames to that egress port is then also congested.

QoS is effective for some configurations but not for others. QoS scheduler (DWRR traffic scheduler) operates within the VOQ of an ingress port independently of other ingress ports. Central arbiter provides an equal chance to each ingress port to talk to an egress port **regardless of priority**. If traffic on one ingress port is of low priority and on another ingress port of high priority, central arbiter will service these queues *equally* when they try to send traffic to the same egress port. Priority starts to matter when one ingress port receives traffic with different priorities destined for the same egress port.

In other words, for QoS to be effective, all competing traffic must enter the switch through the same ingress port X and exit through the same egress port Y. If they exit through different egress ports the congestion does not occur. Once at least one egress port is congested, it causes the ingress port also to be congested thus starving other egress ports. This is the key for understanding QoS on Cisco MDS switches.

QoS on its own will provide better latency for higher priority traffic but it will not offer better bandwidth. Make sure to enable FCC in order to provide better bandwidth for higher priority traffic. For example, Online Transaction Processing (OLTP) traffic is vulnerable to latency spikes but is impervious to bandwidth restrictions. On the other hand, Backup traffic needs a lot of bandwidth and doesn't care about latency fluctuations. With QoS alone, you will only guarantee better latency to OLTP traffic, but the bandwidth will be shared equally between OLTP and Backup traffic. With QoS and FCC enabled together, you will guarantee better latency for OLTP traffic, and provide more bandwidth to backup transactions.

QoS policy configuration steps:

1. Enable QoS on **all switches** in the path
2. Classify traffic only on the **source switch** based on:
    a. source and/or destination FCID,
    b. source and/or destination pWWN,
    c. local source interface,
    d. all (`match any`).
3. Prioritize each traffic class into one of the three queues (High, Medium, or Low) on the **source switch**. Any traffic that was not classified will be put into the default Low queue.
4. Apply QoS service policy to a specific VSAN or VSAN range on the **source switch**
5. Optionally, adjust DWRR weights on **all switches** in the path

On the source switch (the switch with the traffic source attached to), QoS is configured by identifying what traffic should be marked into which of the three queues. Traffic that is not specified falls into the Low queue, by default.

All frames that travel through MDS switches are tagged with an EISL header on the ingress port on the source switch. This behavior cannot be changed. The EISL header is removed when the frame exits the egress port on the destination switch, or when the frame enters a non-trunking ISL link. Part of the EISL header is a 3 bit priority field. It can be set from 0 to 7.

QoS service policy is only responsible for marking traffic on the source switch. Once the frame EISL priority field is marked with a desired priority, it doesn't change all the way from the source to the destination switch as long as all trunks in between are using EISL encapsulation. Priority field is unchanged regardless if QoS is enabled or not on the switches in the middle and on the destination switch. The only way to change EISL priority field is to configure another QoS service policy somewhere in the traffic path.

The actual VOQ queue prioritization doesn't start until QoS is enabled on a particular switch.

By default, when QoS is not enabled, control frames are set with PQ priority 7 and data frames are set with 0. Control Frame EISL priority can be changed to 0 (not recommended). You don't need to have `qos enable` for this behavior to work.

```
switch(config)# qos control priority 0
```

0 is the only number you can specify. If you do this, control traffic will be treated the same as the rest of data traffic. To change back to the default priority of 7, configure:

```
switch(config)# no qos control priority 0
```

When QoS is enabled:

- control frame (PQ) EISL header priority field is set to 7
- high queue data frame EISL header priority field is set to 5
- medium queue data frame EISL header priority field is set to 2
- low queue data frame EISL header priority field is set to 0

These are the only usable values.

EISL trunks (TE ports) are required between MDS switches for QoS to operate.

For unidirectional QoS you don't need to configure QoS policy on the destination switch, just enable QoS on it. When EISL frame enters the ingress TE port on the destination switch, it will be put into the appropriate VOQ based on the priority field in the EISL frame header.

Don't forget about the return path. If you believe that QoS is needed for the return path as well, configure QoS in the reverse direction. Your destination switch is now the source switch, and the source switch is the destination switch.

In other words, if your path consists of three switches, only the source switches require QoS policy configured for traffic from source to destination. The middle switch doesn't need QoS policy configured at all. All three switches must have QoS enabled and should have FCC enabled. You may also change the default DWRR weights on all three switches.

Theory: Configuring QoS

Enable QoS on all switches. If you forget to enable QoS, your VOQs on a switch will stay in dual-queue mode (PQ and Lower Queue).

```
switch(config)# qos enable
```

Classify traffic by creating class-maps and identifying which traffic belongs to which class.

Only one match is needed to classify traffic:

```
switch(config)# qos class-map OLTP match-any
```

All entries must match to classify traffic (this is default):

```
switch(config)# qos class-map OLTP match-all
```

Match based on the source FCID:

```
switch(config-cmap)# match source-address 0x1a2200
switch(config-cmap)# match source-address 0x1a2200 mask 0xffffff
switch(config-cmap)# match source-address 0x1a2200 mask 0xffff00
switch(config-cmap)# match source-address 0x1a0000 mask 0xff0000
```

Match based on the destination FCID:

```
switch(config-cmap)# match destination-address 0x1b2200
switch(config-cmap)# match destination-address 0x1b2200 mask 0xffffff
switch(config-cmap)# match destination-address 0x1b2200 mask 0xffff00
switch(config-cmap)# match destination-address 0x1b0000 mask 0xff0000
```

Match based on the source or destination WWN:

```
switch(config-cmap)# match source-wwn 20:00:00:00:00:00:00:01
switch(config-cmap)# match destination-wwn 20:00:00:00:00:00:00:02
```

Match based on the source or destination Device Alias:

```
switch(config-cmap)# match source-device-alias HOST-A
switch(config-cmap)# match destination-device-alias HOST-B
```

Match based on *local* source interface.

```
switch(config-cmap)# match input-interface fc1/1
```

Create Policy Maps and specify a priority for each traffic class.

```
switch(config)# qos policy-map NewPolicy
switch(config-pmap)# class OLTP
switch(config-pmap-c)# priority high
switch(config-pmap)# class Backup
switch(config-pmap-c)# priority medium
```

Default setting for each class is `priority low`. If you try to configure it, you will get an error `param same as existing one`.

Instead of `priority` command you may also use the `dscp` command.

```
switch(config-pmap-c)# dscp 40
```

Currently, Fibre Channel standard has no specification for DSCP field in the FC header. In the future, you might be able to specify DSCP range from 1 to 63 which will set DSCP field in the FC header. Today, this command is automatically converted into the `priority` command and sets the 3-bit priority field in the EISL header. Refer to the DSCP mappings in Table 50. Note that value 46 may not be used. It's reserved for the priority queue (control traffic).

Table 50: DSCP mapping to EISL priority

DSCP	EISL Header Priority Field (3 bits)	`priority` command	Queue
46	111 (7)	may not be used	Priority Queue (PQ)
10, 12, 14	101 (5)	high	High Queue, DWRR 1
18, 20, 22	010 (2)	medium	Medium Queue, DWRR 2
26, 28, 30	000 (0)	low	Low Queue, DWRR 3
34, 36, 38	000 (0)	low	Low Queue, DWRR 3
any other value	000 (0)	low	Low Queue, DWRR 3

Apply QoS policy map to a VSAN or a range of VSANs.

```
switch(config)# qos service policy NewPolicy vsan 10
switch(config)# qos service policy NewPolicy vsan 10-15
```

QoS can be configured one of two ways, but not both at the same time:

1. Flow-based: apply QoS service policy to a VSAN. If a VSAN has a QoS service policy applied, you can't enable zone-based QoS for a zone that is part of that VSAN.

2. Zone-based: configure QoS attribute for a zone. If a zone in a given VSAN has QoS attribute configured, you can't apply flow-based QoS service policy to that VSAN.

Zone-based QoS configuration is very simple and does not involve class-map and policy-map configurations. You only have to configure `attribute qos` for a zone and activate the zoneset. Traffic between zone members will automatically be prioritized.

```
switch(config)# zone name YourZone vsan 5
switch(config-zone)# attribute qos priority high
switch(config-zone)# attribute qos priority medium
switch(config-zone)# attribute qos priority low
```

If you try to activate a QoS service policy in VSAN 5, you'll get an error:

```
switch(config)# qos service policy YourPolicy vsan 5
Failed to effect configuration: Active Zoneset has QoS Attribute
```

QoS service policy is applied to an individual VSAN. The actual QoS prioritization happens globally for all VSANs combined. VOQs are not VSAN-aware. Your High, Medium, and Low VOQs are shared between all VSANs in a given ingress port.

One key difference between VSAN-based (flow-based) and Zone-based QoS:

1. If the same FC flow is referenced in two class maps associated to a policy map, the priority value of the class map attached first will take effect.

2. If the same FC flow exists in two zones and they have different QoS attributes configured, the higher priority attribute will be used.

Optionally, you can adjust DWRR weights. Keep in mind, that DWRR weights configuration affects all QoS policies configured for all VSANs.

```
switch(config)# qos dwrr-q high weight 70
switch(config)# qos dwrr-q medium weight 15
switch(config)# qos dwrr-q low weight 15
```

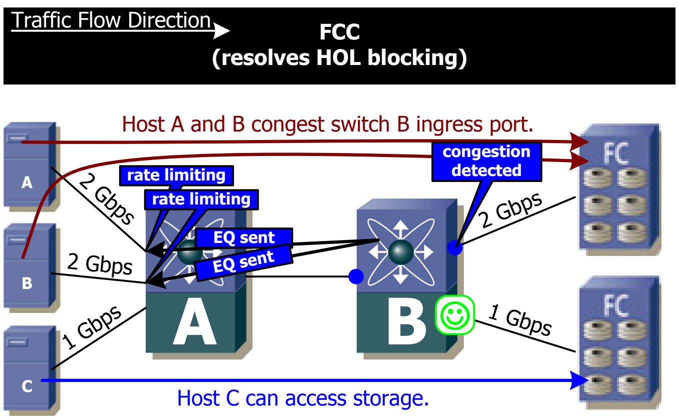
To verify QoS configuration, use the following commands:

```
switch# show qos class-map
switch# show qos policy-map
switch# show qos dwrr
switch# show qos service policy
switch# show qos statistics
```

Theory: FCC

FCC (Fibre Channel Congestion Control) detects and reacts to a network congestion situation, avoiding the HOL blocking. FCC is disabled, by default. It is part of free package since SAN-OS 1.0(2).

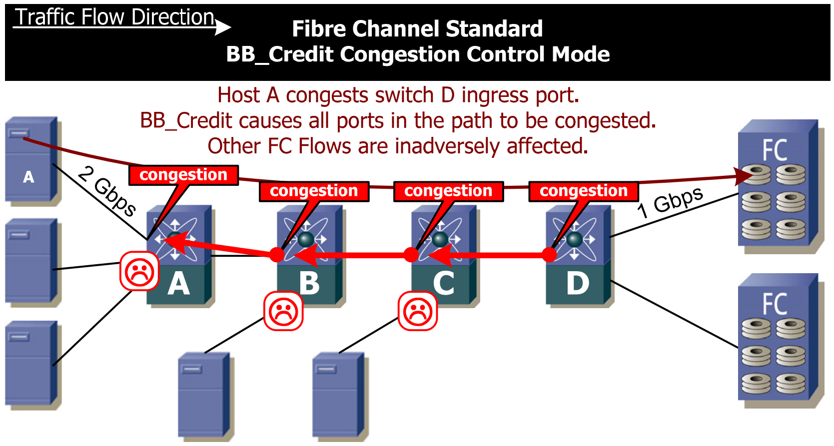
Figure 237: FCC



1. Congestion is detected on an egress port on a destination switch
2. A special frame called Edge Quench (EQ) is sent from the destination switch to the source switch
3. The switch that source is attached to intercepts EQ frame. It then starts to rate limit incoming traffic on the ingress port that the source is attached to. It does it by slowing down the rate of R_RDY primitives sent back to the source. Essentially, it cuts B2B credits in half. Each EQ frame rate limits traffic by 50%. It is not necessary to undo this action because traffic eventually builds up again.

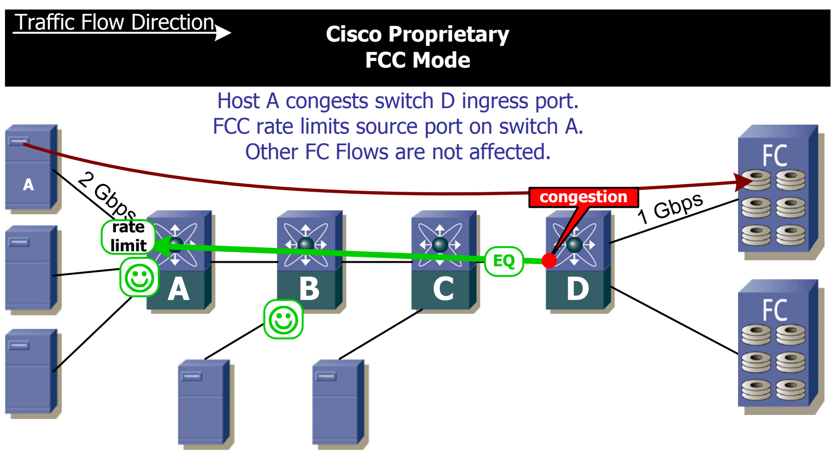
Without FCC the switches are left to use the buffer-to-buffer (B2B) credit mode, the only flow-control mechanism available for Fibre Channel Class 3 frames. Buffer-to-buffer credits are negotiated on a hop-by-hop basis. BB_Credit flow control is ineffective in the real world scenarios. When an ingress port is congested on the destination switch, the port will move to BB_Credit mode. A credit (R_RDY) is returned to the source port, only when the ingress port is capable of sending the frame to the egress port. Because the port is congested, credits are not sent thus making the source port congested as well. With BB_Credit mode, congestion on the destination switch will have a ripple effect and congest all ports in the path back to the source.

Figure 238: FC standard BB_Credit congestion control



With FCC enabled, if congestion occurs on the destination switch, it will send the Edge Quench message all the way to the source of the traffic. The source switch intercepts the frame and starts to rate limit the source of the traffic, thus preventing congestion in the entire path!

Figure 239: FCC in operation



FCC EQ frame can travel through non-Cisco fabric switches because it is a regular FC frame. *The source and destination switches* must be Cisco MDS Switches in order for FCC to operate.

FCC essentially controls bandwidth. FCC is easy to configure. If you decide to use it, make sure to enable it on every switch in your fabric.

```
switch(config)# fcc enable
```

By default, FCC will rate limit the frames with EISL header priority in the range of 0-4.

1. If QoS is disabled, FCC will by default rate limit all data traffic (0) and will not rate limit control traffic (7)

2. If QoS is enabled, FCC will by default rate limit low (0) and medium (2) priority traffic and will not rate limit high (5) and control traffic (7)

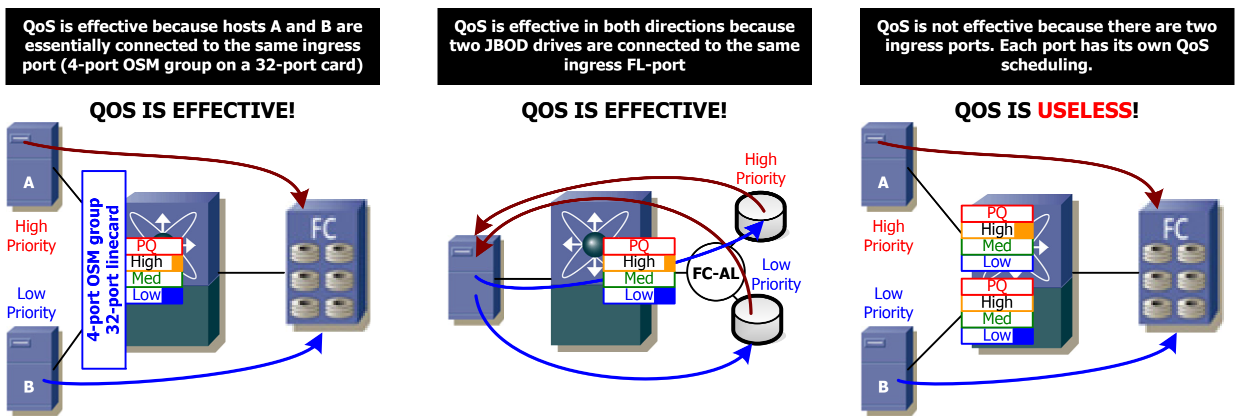
You can change this behavior by increasing or decreasing FCC priority. For example, if you want to prevent FCC from rate limiting medium (2) priority traffic, use the following command:

```
switch(config)# fcc priority 1
```

Now, only low (0) priority traffic will be rate limited.

Let's look at some scenarios where QoS is effective and where it is useless. Recall that **for QoS to be effective, all competing traffic must enter the switch through the same ingress port X and exit through the same egress port Y.** This requirement leads to another requirement that you need to have at least two switches connected with an EISL trunk for QoS to be effective. However, there are some cases where QoS can be effective in a single switch fabric.

Figure 240: QoS in single switch topologies



MDS91x0, MDS9216i and MPS-14/2 module offer Ingress Port Rate Limiting feature. It is not supported on FC16 and FC32 modules. This feature controls ingress traffic into an FC port. This feature operates similarly to FCC rate limiting by slowing down the rate of R_RDY frames.

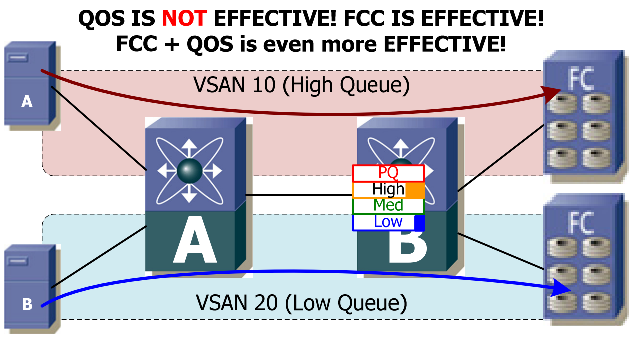
```
switch(config)# interface fc1/1
switch(config-if)# switchport ingress-rate 75
```

The above command configures 75% ingress port rate.

Can you prioritize one VSAN over another VSAN using QoS? Here is a possible solution, but it's not very effective:

1. create two class-maps, one for each VSAN, with `match any` statement
2. create two policy-maps, one for each class (VSAN), with the appropriate priority setting (high, medium, or low)
3. apply each policy-map to a respective VSAN

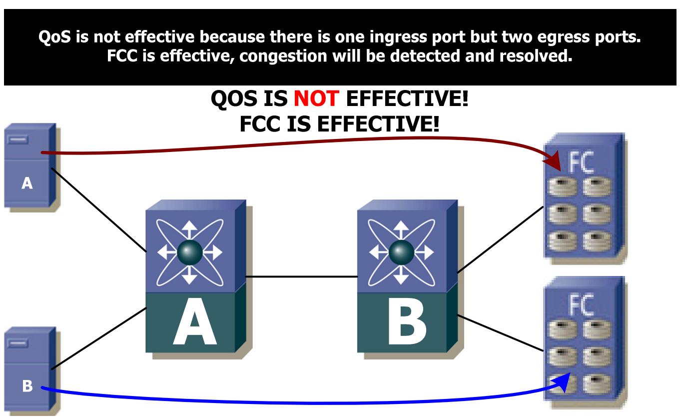
Figure 241: Prioritizing one VSAN over another



This QoS configuration would not be effective. If you have two VSANs, you would then have two separate FC Flows, two different sources talking to two different destinations. The destination switch, would have a common ingress port (TE EISL trunk), but two different egress ports. Do you remember our key statement? **For QoS to be effective, all competing traffic must enter the switch through the same ingress port X and exit through the same egress port Y.**

The better solution is to use FCC. FCC would resolve the congestion issue.

Figure 242: FCC is more effective than QoS in multi-egress interface topology



Keep in mind that, by default, FCC priority is set to 4. FCC would only apply to Low and Medium traffic. FCC would not try to rate limit High queue traffic.

Theory: QoS on Generation-2 modules

Starting with SAN-OS 3.0(1), Generation-2 modules introduce a slight change in DWRR behavior. QoS DWRR in Gen-1 modules affects traffic coming in through a given port and queued to the same egress port. QoS DWRR in Gen-2 modules affects traffic coming in through a given port and queued to **any egress ports**. Let's make an additional statement. **For QoS to be effective, all competing traffic must enter the switch through the same ingress Gen-2 port X and exit through any egress Gen-2 port.** Traffic from Gen-1 port to Gen-1 port, from Gen-1 port to Gen-2 port, from Gen-2 port to Gen-1 port is not affected by this.

Walkthrough Solutions

Task 17.1

By default, MDS switches use standard Buffer-to-Buffer credit mode for congestion detection. BB_Credit mode is not as effective as Fibre Channel Congestion control mode (FCC). By default, FCC will detect and rate limit congestion for traffic with EISL header priority in the range of 0-4. Only "low (0)" and "medium (2)" traffic falls into this range. The requirement is to make sure that "high (5)" traffic is also controlled by FCC.

Control traffic between switches is automatically marked with EISL header priority of 7. We are asking you not to control "absolute priority queue (7)" traffic.

Make sure to enable FCC on all switches.

```
MDS1(config)# fcc enable
MDS1(config)# fcc priority 5

MDS2(config)# fcc enable
MDS2(config)# fcc priority 5

MDS3(config)# fcc enable
MDS3(config)# fcc priority 5
```

Task 17.2

Control traffic between switches is automatically marked with EISL header priority of 7. This traffic ends up in the absolute priority queue which is always serviced first whether or not QoS is enabled. You can turn off automatic marking of inter-switch control traffic. This will cause all inter-switch control traffic to have 0 in the EISL header priority field.

```
MDS1(config)# qos control priority 0

MDS2(config)# qos control priority 0

MDS3(config)# qos control priority 0
```

This change is not recommended in production environment.

Task 17.3

Recall that for QoS to be effective all competing traffic must enter the switch through the same ingress port X and exit through the same egress port Y. They could exit through different egress ports, but then congestion wouldn't occur. Once at least one egress port is congested, it causes the ingress port to also become congested, thus starving other egress ports.

In addition, FCC is not effective if the source and the destination are connected to the same switch. QoS is also generally not effective for devices attached to the same switch because it relies on two or more devices sharing the same ingress port.

In our case ingress traffic comes in on different interfaces on MDS1. Traffic from J2B1 to WIN1A comes in on ISL trunk fc1/14 and traffic from J1A1 to WIN1A comes in on F_Port fc1/4. There is nothing you can do to give one traffic better preference over the other. The central arbiter services all input ports (and all of their queues) equally regardless of QoS configuration.

Remember that for QoS to be effective all competing traffic must enter the switch through the same ingress port X.

There is no solution for this task.

Task 17.4

QoS can be used in this task because competing traffic would use the same ingress port. Let's examine this a little closer. WIN2B and WIN3B send traffic across the fabric to J3A drives. This traffic could flow from MDS2 to MDS3 directly or via MDS1. This simply depends on current FSPF costs and failover status.

Congestion is possible on MDS3 fc1/4 interface if combined traffic rate from WIN2 and WIN3 overrun buffers on JBOD3 Port A. This is determined by buffer-to-buffer credits returned by J3A to MDS3 fc1/4. Once the egress port fc1/4 is congested, the ingress port will also be congested. If traffic arrives on one ISL trunk, for example MDS3 fc1/3, then fc1/3 will become congested. If traffic arrives on more than one ISL trunk, for example MDS3 fc1/3 and Po2 then there's a smaller chance of congestion because there's more available capacity on the ingress interfaces. This really all depends on load-balancing mechanism in place and current congestion levels on the interfaces.

If ingress port is congested, QoS can help to provide for better latency to preferred traffic over less preferred traffic. We are mentioning latency first because that is the first effect of QoS configuration. The higher priority queue is serviced more often than the lower priority queue thus providing for a better average delay to higher priority traffic. If we talk about only two sources of traffic then their bandwidth will be split in half regardless of DWRR queue ratios. This behavior is normal when FCC is not enabled (default). If FCC is enabled, then bandwidth treatment will also go in effect. Higher priority traffic will have more bandwidth available than lower priority traffic.

In our case FCC is already enabled. If FCC wasn't enabled, we would have to enable it for this task. If the question didn't mention "bandwidth" and only mentioned "average delay" or "latency" then we wouldn't be required to enable FCC.

QoS is much more effective with FCC enabled than with FCC not enabled.

Our first task is to make sure WIN2 to J3A traffic ends up in a higher priority queue than WIN3 to J3A traffic. Also, we want WIN3 to J3A traffic to end up in a higher priority queue than all other traffic. We can use high priority queue for WIN2 to J3A traffic, medium priority queue for WIN3 to J3A traffic, and low priority queue for all other traffic. All traffic by default is placed in the low priority queue already.

In order for traffic to be placed in a high priority queue on a given ingress interface it must be marked with a 5 in the EISL Header Priority field. Medium priority traffic must be marked with a 2, and low priority traffic will by default have 0 in the EISL header priority field.

We must configure QoS service policy to mark traffic appropriately. We are asked to configure QoS policy on one switch only. We must do this on MDS2, the switch connected to the source of the traffic. We don't need to configure QoS policy on MDS1 and MDS3 because EISL header priority field will be preserved when traffic flows through the entire fabric all the way to the destination. You must enable QoS on all three switches in order for QoS to queue traffic appropriately based on markings.

It's important to note that this configuration only takes care of traffic in one direction from servers to storage. Our question explicitly mentions this direction only. If the question didn't mention the direction of traffic then you would need to configure QoS for each direction of traffic. You would configure QoS in one direction on the source switch MDS2 and in another direction on the other source switch MDS3.

We need to carefully choose how we are going to match traffic for QoS marking. We ask you to make sure this configuration is not affected by the Domain ID of the MDS switches. This means that we can't use FCID for matching of traffic. We also should not use `input-interface` for matching because we care what destination the traffic goes to. We should use `source-wwn` or `source-device-alias` for matching the source of the traffic. We also need to match on the destination. In our case we have not explicitly specified what JBOD3 drives are involved. Therefore, we need to match on `destination-wwn` or `destination-device-alias` of each J3A drive. QoS class-maps allow for `match-any` or `match-all` (default) matching. Either option can't help us configure everything in one class-map. We must configure a separate class-map for each source/destination pair. We will leave the default `match-all` matching criteria.

Enable QoS on all three switches.

```
MDS1(config)# qos enable

MDS2(config)# qos enable

MDS3(config)# qos enable
```

Configure QoS class-maps on MDS2.

```
MDS2(config)# qos class-map WIN2B-J3A1
MDS2(config-cmap)# match source-device-alias WIN2B
Device-Alias enhanced mode is not supported by QoS
```

QoS cannot be used with enhanced Device Aliases. We have to use WWN instead.

```
MDS2# show device-alias name WIN2B
device-alias name WIN2B pwwn 21:01:00:e0:8b:29:f5:46
MDS2# show device-alias name WIN3B
device-alias name WIN3B pwwn 21:01:00:e0:8b:29:93:46
MDS2# show device-alias name J3A1
device-alias name J3A1 pwwn 21:00:00:20:37:09:77:f6
MDS2# show device-alias name J3A2
device-alias name J3A2 pwwn 21:00:00:20:37:0d:d9:05
MDS2# show device-alias name J3A3
device-alias name J3A3 pwwn 21:00:00:20:37:16:82:24
MDS2# show device-alias name J3A4
device-alias name J3A4 pwwn 21:00:00:20:37:0e:af:5f
MDS2# show device-alias name J3A5
device-alias name J3A5 pwwn 21:00:00:20:37:0e:89:b4
MDS2# show device-alias name J3A6
device-alias name J3A6 pwwn 21:00:00:20:37:09:3f:7c

MDS2(config)# qos class-map WIN2B-J3A1
MDS2(config-cmap)# match source-wwn 21:01:00:e0:8b:29:f5:46
Operation in progress. Please check class-map parameters
MDS2(config-cmap)# match destination-wwn 21:00:00:20:37:09:77:f6
Operation in progress. Please check class-map parameters
MDS2(config-cmap)# exit
MDS2(config)# qos class-map WIN2B-J3A2
MDS2(config-cmap)# match source-wwn 21:01:00:e0:8b:29:f5:46
Operation in progress. Please check class-map parameters
MDS2(config-cmap)# match destination-wwn 21:00:00:20:37:0d:d9:05
Operation in progress. Please check class-map parameters
MDS2(config-cmap)# exit
MDS2(config)# qos class-map WIN2B-J3A3
MDS2(config-cmap)# match source-wwn 21:01:00:e0:8b:29:f5:46
Operation in progress. Please check class-map parameters
MDS2(config-cmap)# match destination-wwn 21:00:00:20:37:16:82:24
Operation in progress. Please check class-map parameters
MDS2(config-cmap)# exit
MDS2(config)# qos class-map WIN2B-J3A4
MDS2(config-cmap)# match source-wwn 21:01:00:e0:8b:29:f5:46
Operation in progress. Please check class-map parameters
MDS2(config-cmap)# match destination-wwn 21:00:00:20:37:0e:af:5f
Operation in progress. Please check class-map parameters
MDS2(config-cmap)# exit
MDS2(config)# qos class-map WIN2B-J3A5
MDS2(config-cmap)# match source-wwn 21:01:00:e0:8b:29:f5:46
Operation in progress. Please check class-map parameters
MDS2(config-cmap)# match destination-wwn 21:00:00:20:37:0e:89:b4
Operation in progress. Please check class-map parameters
MDS2(config-cmap)# exit
MDS2(config)# qos class-map WIN2B-J3A6
```

```
MDS2(config-cmap)# match source-wwn 21:01:00:e0:8b:29:f5:46
Operation in progress. Please check class-map parameters
MDS2(config-cmap)# match destination-wwn 21:00:00:20:37:09:3f:7c
Operation in progress. Please check class-map parameters
MDS2(config-cmap)# exit
MDS2(config)# qos class-map WIN3B-J3A1
MDS2(config-cmap)# match source-wwn 21:01:00:e0:8b:29:93:46
Operation in progress. Please check class-map parameters
MDS2(config-cmap)# match destination-wwn 21:00:00:20:37:09:77:f6
Operation in progress. Please check class-map parameters
MDS2(config-cmap)# exit
MDS2(config)# qos class-map WIN3B-J3A2
MDS2(config-cmap)# match source-wwn 21:01:00:e0:8b:29:93:46
Operation in progress. Please check class-map parameters
MDS2(config-cmap)# match destination-wwn 21:00:00:20:37:0d:d9:05
Operation in progress. Please check class-map parameters
MDS2(config-cmap)# exit
MDS2(config)# qos class-map WIN3B-J3A3
MDS2(config-cmap)# match source-wwn 21:01:00:e0:8b:29:93:46
Operation in progress. Please check class-map parameters
MDS2(config-cmap)# match destination-wwn 21:00:00:20:37:16:82:24
Operation in progress. Please check class-map parameters
MDS2(config-cmap)# exit
MDS2(config)# qos class-map WIN3B-J3A4
MDS2(config-cmap)# match source-wwn 21:01:00:e0:8b:29:93:46
Operation in progress. Please check class-map parameters
MDS2(config-cmap)# match destination-wwn 21:00:00:20:37:0e:af:5f
Operation in progress. Please check class-map parameters
MDS2(config-cmap)# exit
MDS2(config)# qos class-map WIN3B-J3A5
MDS2(config-cmap)# match source-wwn 21:01:00:e0:8b:29:93:46
Operation in progress. Please check class-map parameters
MDS2(config-cmap)# match destination-wwn 21:00:00:20:37:0e:89:b4
Operation in progress. Please check class-map parameters
MDS2(config-cmap)# exit
MDS2(config)# qos class-map WIN3B-J3A6
MDS2(config-cmap)# match source-wwn 21:01:00:e0:8b:29:93:46
Operation in progress. Please check class-map parameters
MDS2(config-cmap)# match destination-wwn 21:00:00:20:37:09:3f:7c
Operation in progress. Please check class-map parameters
```

Ignore the MDS warning `Operation in progress. Please check class-map parameters`.

Next, configure QoS policy-map on MDS2. We ask you not to use `priority` keyword when configuring QoS policy map. Instead, we can use `DSCP` keyword to achieve the same result.

You must use DSCP values 10, 12 or 14 for high priority and values 18, 20 or 22 for medium priority.

```
MDS2(config)# qos policy-map qos-map-vsan20
MDS2(config-pmap)# class WIN2B-J3A1
MDS2(config-pmap-c)# dscp 10
Operation in progress. Please check class-map parameters
MDS2(config-pmap-c)# class WIN2B-J3A2
MDS2(config-pmap-c)# dscp 10
Operation in progress. Please check class-map parameters
MDS2(config-pmap-c)# class WIN2B-J3A3
MDS2(config-pmap-c)# dscp 10
Operation in progress. Please check class-map parameters
MDS2(config-pmap-c)# class WIN2B-J3A4
MDS2(config-pmap-c)# dscp 10
Operation in progress. Please check class-map parameters
MDS2(config-pmap-c)# class WIN2B-J3A5
MDS2(config-pmap-c)# dscp 10
Operation in progress. Please check class-map parameters
MDS2(config-pmap-c)# class WIN2B-J3A6
MDS2(config-pmap-c)# dscp 10
Operation in progress. Please check class-map parameters
MDS2(config-pmap-c)# class WIN3B-J3A1
MDS2(config-pmap-c)# dscp 18
Operation in progress. Please check class-map parameters
MDS2(config-pmap-c)# class WIN3B-J3A2
MDS2(config-pmap-c)# dscp 18
Operation in progress. Please check class-map parameters
MDS2(config-pmap-c)# class WIN3B-J3A3
MDS2(config-pmap-c)# dscp 18
Operation in progress. Please check class-map parameters
```

```
MDS2(config-pmap-c)# class WIN3B-J3A4
MDS2(config-pmap-c)# dscp 18
Operation in progress. Please check class-map parameters
MDS2(config-pmap-c)# class WIN3B-J3A5
MDS2(config-pmap-c)# dscp 18
Operation in progress. Please check class-map parameters
MDS2(config-pmap-c)# class WIN3B-J3A6
MDS2(config-pmap-c)# dscp 18
Operation in progress. Please check class-map parameters
```

Ignore the MDS warning `Operation in progress. Please check class-map parameters`.

Verify the QoS class-map and policy-map configurations. `DSCP` keyword is automatically replaced with the `priority` keyword in the configuration.

```
MDS2# show qos class-map
qos class-map test match-all
qos class-map WIN2B-J3A1 match-all
    match source-wwn 21:01:00:e0:8b:29:f5:46 [WIN2B]
    match destination-wwn 21:00:00:20:37:09:77:f6 [J3A1]
qos class-map WIN2B-J3A2 match-all
    match source-wwn 21:01:00:e0:8b:29:f5:46 [WIN2B]
    match destination-wwn 21:00:00:20:37:0d:d9:05 [J3A2]
qos class-map WIN2B-J3A3 match-all
    match source-wwn 21:01:00:e0:8b:29:f5:46 [WIN2B]
    match destination-wwn 21:00:00:20:37:16:82:24 [J3A3]
qos class-map WIN2B-J3A4 match-all
    match source-wwn 21:01:00:e0:8b:29:f5:46 [WIN2B]
    match destination-wwn 21:00:00:20:37:0e:af:5f [J3A4]
qos class-map WIN2B-J3A5 match-all
    match source-wwn 21:01:00:e0:8b:29:f5:46 [WIN2B]
    match destination-wwn 21:00:00:20:37:0e:89:b4 [J3A5]
qos class-map WIN2B-J3A6 match-all
    match source-wwn 21:01:00:e0:8b:29:f5:46 [WIN2B]
    match destination-wwn 21:00:00:20:37:09:3f:7c [J3A6]
qos class-map WIN3B-J3A1 match-all
    match source-wwn 21:01:00:e0:8b:29:93:46 [WIN3B]
    match destination-wwn 21:00:00:20:37:09:77:f6 [J3A1]
qos class-map WIN3B-J3A2 match-all
    match source-wwn 21:01:00:e0:8b:29:93:46 [WIN3B]
    match destination-wwn 21:00:00:20:37:0d:d9:05 [J3A2]
qos class-map WIN3B-J3A3 match-all
    match source-wwn 21:01:00:e0:8b:29:93:46 [WIN3B]
    match destination-wwn 21:00:00:20:37:16:82:24 [J3A3]
qos class-map WIN3B-J3A4 match-all
    match source-wwn 21:01:00:e0:8b:29:93:46 [WIN3B]
    match destination-wwn 21:00:00:20:37:0e:af:5f [J3A4]
qos class-map WIN3B-J3A5 match-all
    match source-wwn 21:01:00:e0:8b:29:93:46 [WIN3B]
    match destination-wwn 21:00:00:20:37:0e:89:b4 [J3A5]
qos class-map WIN3B-J3A6 match-all
    match source-wwn 21:01:00:e0:8b:29:93:46 [WIN3B]
    match destination-wwn 21:00:00:20:37:09:3f:7c [J3A6]

MDS2# show qos policy-map
qos policy-map qos-map-vsan20
    class WIN2B-J3A1
        priority high
    class WIN2B-J3A2
        priority high
    class WIN2B-J3A3
        priority high
    class WIN2B-J3A4
        priority high
    class WIN2B-J3A5
        priority high

    class WIN2B-J3A6
        priority high
    class WIN3B-J3A1
        priority medium
    class WIN3B-J3A2
        priority medium
    class WIN3B-J3A3
        priority medium
    class WIN3B-J3A4
        priority medium
```

```
    class WIN3B-J3A5
        priority medium
    class WIN3B-J3A6
        priority medium
```

Apply this policy-map to VSAN 20 on MDS2.

```
MDS2(config)# qos service policy qos-map-vsan20 vsan 20
Operation in progress. Please check policy-map parameters

MDS2# show qos service policy
qos service policy qos-map-vsan20 vsan 20
```

By default, high priority queue will be serviced 50% of the time, medium priority queue will be serviced 30% of the time and low priority queue will be services 20% of the time. These ratios are determined by the DWRR weights. When configuring DWRR, make sure that they add up to 100. Also, make sure to configure them the same on all MDS switches in the fabric.

In our case, we want WIN2-JBOD3 traffic to have 2:1 ratio against WIN3-JBOD3 traffic, and WIN3-JBOD3 traffic to have 2:1 ratio against all other traffic. In other words, we want to have 4:2:1 ratio for high, medium and low traffic. How can we calculate DWRR weights to match this ratio? The best way is to find common denominator by adding ratios and dividing 100 by their sum. 4 + 2 + 1 = 7. 100 / 7 = 14. Next, multiply each ratio by 14. 14 x 4 = 56. 14 x 2 = 28. 14 x 1 = 14. Add them together 56 + 28 + 14 and we get 98. You can leave them as is or you can add 1 to, for example, high and medium queues to make the sum equal to 100.

```
MDS1(config)# qos dwrr-q high weight 56
Operation in progress. Please check DWRR parameters
MDS1(config)# qos dwrr-q medium weight 28
Operation in progress. Please check DWRR parameters
MDS1(config)# qos dwrr-q low weight 14
Operation in progress. Please check DWRR parameters

MDS2(config)# qos dwrr-q high weight 56
Operation in progress. Please check DWRR parameters
MDS2(config)# qos dwrr-q medium weight 28
Operation in progress. Please check DWRR parameters
MDS2(config)# qos dwrr-q low weight 14
Operation in progress. Please check DWRR parameters

MDS3(config)# qos dwrr-q high weight 56
Operation in progress. Please check DWRR parameters
MDS3(config)# qos dwrr-q medium weight 28
Operation in progress. Please check DWRR parameters
MDS3(config)# qos dwrr-q low weight 14
Operation in progress. Please check DWRR parameters

MDS3# show qos dwrr
qos dwrr-q high weight 56
qos dwrr-q medium weight 28
qos dwrr-q low weight 14
```

In the lab, you should ask the proctor if you must make the sum of DWRR weights equal to 100 or if they want you to have precise ratios.

Task 17.5

We need to configure QoS policy for this traffic. If FCC was not previously enabled, we wouldn't have to enable it because we are concerned about *average delay*. If we mentioned *bandwidth* we would need to enable FCC also.

Make sure WIN1B and J2A1 are not connected to the same switch, because QoS would bring no value then.

If source and destination are connected to the same switch you should ask your proctor if you should still configure QoS because it would not be effective.

Our task doesn't mention specific direction of the traffic, therefore traffic in both directions should be accounted for. We choose to mark WIN1B to J2A1 traffic with high priority on MDS2, and mark J2A1 to WIN1B traffic with high priority on MDS1. We are not mentioning anything about DWRR weights, therefore we'll leave them as they are.

You shouldn't have two questions in your lab asking for different DWRR weights.

```
MDS2# show device-alias name WIN1B
device-alias name WIN1B pwwn 21:01:00:e0:8b:29:e3:4d
MDS2# show device-alias name J2A1
device-alias name J2A1 pwwn 21:00:00:20:37:09:b6:e0

MDS2(config)# qos class-map WIN1B-J2A1
MDS2(config-cmap)# match source-wwn 21:01:00:e0:8b:29:e3:4d
Operation in progress. Please check class-map parameters
MDS2(config-cmap)# match destination-wwn 21:00:00:20:37:09:b6:e0
Operation in progress. Please check class-map parameters
MDS2(config-cmap)# exit
MDS2(config)# qos policy-map qos-map-vsan30
MDS2(config-pmap)# class WIN1B-J2A1
MDS2(config-pmap-c)# priority high
Operation in progress. Please check class-map parameters
MDS2(config-pmap-c)# exit
MDS2(config-pmap)# exit
MDS2(config)# qos service policy qos-map-vsan30 vsan 30
Operation in progress. Please check policy-map parameters

MDS1(config)# qos class-map J2A1-WIN1B
MDS1(config-cmap)# match source-wwn 21:00:00:20:37:09:b6:e0
Operation in progress. Please check class-map parameters
MDS1(config-cmap)# match destination-wwn 21:01:00:e0:8b:29:e3:4d
Operation in progress. Please check class-map parameters
MDS1(config-cmap)# exit
MDS1(config)# qos policy-map qos-map-vsan30
MDS1(config-pmap)# class J2A1-WIN1B
MDS1(config-pmap-c)# priority high
Operation in progress. Please check class-map parameters
MDS1(config-pmap-c)# exit
MDS1(config-pmap)# exit
MDS1(config)# qos service policy qos-map-vsan30 vsan 30
Operation in progress. Please check policy-map parameters
```

Task 17.6

This task is similar to the previous one, except that we cannot configure QoS class-map on MDS1. If we can't configure class-map on MDS1 to match WIN1A-J2B1 traffic that we can't mark this traffic with desired high priority. If we can't mark the traffic at the source, then QoS will not work.

Instead of using VSAN-based QoS policy we can use zone-based QoS policy. They are mutually exclusive, you can't have both configured in one VSAN.

We have already configured Zone-based QoS in a zone lab.

There is a difference in configuring zone-based QoS for normal and enhanced zoning. In normal zoning you simply add QoS parameter to the zone and in enhanced zoning you need to use zone attribute groups. VSAN 10 is running normal zoning.

Do you remember that full zoneset distribution is not enabled in VSAN 10? We stressed this point back in Zoning lab. You need to copy active zoneset to full zoneset on MDS2 before making changes to zoning in VSAN 10.

```
MDS1# zone copy active-zoneset full-zoneset vsan 10
WARNING: This command may overwrite common zones
         in the full zoneset
Please enter yes to proceed.(y/n) [n]? y

MDS1# show zoneset active vsan 10
zoneset name VSAN10 vsan 10
<output omitted>
  zone name WIN1A-J2B1 vsan 10
  * fcid 0x200000 [device-alias WIN1A]
  * fcid 0x3200aa [device-alias J2B1]
<output omitted>

MDS1# conf t
Enter configuration commands, one per line.  End with CNTL/Z.
MDS1(config)# zone name WIN1A-J2B1 vsan 10
MDS1(config-zone)# attribute qos priority high
MDS1(config-zone)# exit
MDS1(config)# zoneset activate name VSAN10 vsan 10
Zoneset activation initiated. check zone status
```

Save configurations before proceeding.

```
MDS1# wr
[########################################] 100%

MDS2# wr
[########################################] 100%

MDS3# wr
[########################################] 100%
```