Lossless Network Deployment with iSCSI for SAN Fabric on Air Gap Core with Multipath using LLFC and PFC

Open Source Code

This product includes code licensed under certain open source licenses which require source compliance. The corresponding source for these components is available upon request. This offer is valid to anyone in receipt of this information and shall expire three years following the date of the final distribution of this product version by Hewlett Packard Enterprise Company. To obtain such source code, please check if the code is available in the HPE Software Center at https://myenterpriselicense.hpe.com/cwp-ui/software but, if not, send a written request for specific software version and product for which you want the open source code. Along with the request, please send a check or money order in the amount of US $10.00 to:

Hewlett Packard Enterprise Company
Attn: General Counsel
WW Corporate Headquarters
1701 E Mossy Oaks Rd, Spring, TX 77389
United States of America

Contents

Revision History

The following table lists the revisions of this document.

Table 1: *Revision History*

Revision	Change Description
Revision 01	Initial release.
Revision 02	Added HPE ANW CX 8325H-16C and CX 8325H-18Y4C Switches.

About this Guide

This guide provides information on deploying HPE Aruba Networking (HPE ANW) CX Switches for iSCSI multipath solution. It also provides information about Multipath setup and iSCSI deployment on a Red Hat Enterprise Linux host.

Intended Audience

This guide is intended for users managing HPE Aruba Networking CX Data Center switches in the deployment of an iSCSI networking solution.

Applicable Products

This document applies to the following products:

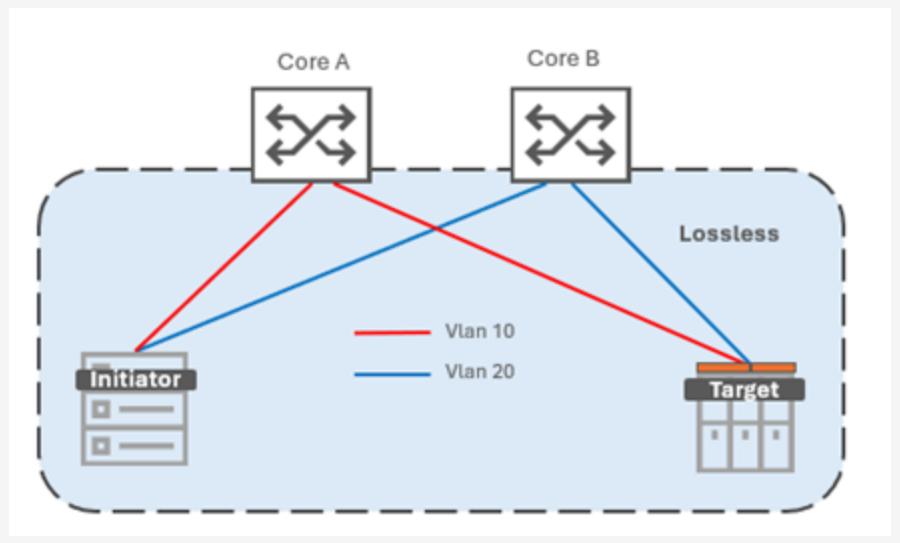
- HPE Aruba Networking CX 8325-32C Switch
- HPE Aruba Networking CX 8325H-16C Switch
- HPE Aruba Networking CX 8325H-18Y4C Switch
- HPE Aruba Networking CX 9300S Switch
- HPE Aruba Networking CX 9300-32D Switch

Overview of Air Gap for iSCSI with Multipath using LLFC or PFC

The Internet Small Computer Systems Interface (iSCSI) network can function as either a dedicated network or one shared with traditional Ethernet applications. Notably, the iSCSI traffic component is designed to be lossless. Data storage represents a critical asset for any organization, underscoring the paramount importance of maintaining data integrity and reliability.

The following figure illustrates a topology that uses two HPE Aruba Networking (HPE ANW) CX switches to provide connectivity to a Host Initiator (running Linux as the operating system) and an iSCSI Target. To provide high availability, the Device Mapper Multipath is used to build two paths between Target and Initiator, where each path has its own network and a single VLAN. This is a Layer 2 design that includes two standalone switches, including implementation of Link Layer Flow Control (LLFC) or Priority-based Flow Control (PFC).

Figure 1 *Air Gap Core Topology for iSCSI*



To deploy a lossless network on the HPE ANW CX switches, a set of QoS components must be applied for packet Prioritization, Classification, Queueing, and Transmission. Both the flow control protocols are used in these scenarios - Link Level Flow Control (LLFC) or Priority Flow Control (PFC).

To get a complete solution, all devices in the network must be configured to use, accept, or setup QoS settings to get full lossless path for storage traffic. The initiator interfaces and target interfaces are 100G Ethernet Smart NICs. Each switch manages a path to the Target in an active-standby mode or active-active mode, governed by the Multipath feature.

Configuring Lossless Network on HPE Aruba Networking CX Switches

To setup a lossless network on the HPE ANW CX switches, complete the following configuration steps:

1. Configuring QoS Classes and Policies
2. Configuring QoS Trust and MTU
3. Configuring QoS Queue and Schedule Profile
4. Configuring QoS Pool
5. Configuring LLFC
6. Configuring PFC
7. Configuring PFC with DCBX

Configuring QoS Classes and Policies

The HPE ANW CX operating system classifies and marks storage traffic to be handled with lossless priorities. An IPv4 or IPv6 class identifies match criteria that can be used in policy to remark traffic with the appropriate Quality of Service (QoS) priority. The **ip class** stanza matches traffic on standard IP, TCP, and UDP header values in addition to several other fields.

The following configuration **tcp_iscsi_initiator** class matches TCP traffic from any source to any destination IP, but only for the iSCSI destination protocol port 3260.

```
class ip tcp_iscsi_initiator
  10 match tcp any any eq iscsi count
```

The following configuration **v6_tcp_iscsci_initiator** class is for IPv6 class.

```
class ipv6 v6_tcp_iscsi_initiator
  10 match tcp any any eq iscsi count
```

A policy remarks traffic matching a specified IP class with the desired QoS priority. In the following policy example, all traffic matching the **tcp_iSCSI_initiator** IP class is remarked with **PCP 4**, **DSCP CS4**, and assigned a **local priority of 4**.

```
policy remark_iSCSI_Initiator
  10 class ip tcp_iscsi_initiator action pcp 4 action local-priority 4 action dscp
CS4
  20 class ipv6 v6_tcp_iscsi_initiator action pcp 4 action local-priority 4 action
dscp CS4
```

The **local-priority** value is used internally by the switch and maps traffic to corresponding queues. There are eight local priority values that can be used to prioritize traffic. In the policy configuration example, the iSCSI traffic is remarked to **local priority 4**, **PCP 4** and **DSCP CS4**.

In the following example, IP classes **tcp_iscsi_target** and **v6_tcp_iscsi_target** are defined for IPv4 and IPv6 respectively. The **tcp_iscsi_target** and **v6_tcp_iscsi_target** classes match the traffic sourced by the target, and then proceed with remarking using policy **remark_iSCSI_Target**.

The packets matching an TCP source port with iSCSI (3260) are remarked with the same QoS priorities as traffic sourced by the Initiator. This must be applied ingress on the interface connected to a target.

```
class ip tcp_iscsi_target
  10 match tcp any eq iscsi any count
class ipv6 v6_tcp_iscsi_target
  10 match tcp any eq iscsi any count
…
…
policy remark_iSCSI_Target
  10 class ip tcp_iscsi_target action pcp 4 action local-priority 4 action
dscp CS4
  20 class ipv6 v6_tcp_iscsi_target action pcp 4 action local-priority 4
action dscp CS4
```

Use the **show policy hitcounts** command to display hit counts against the class in policy applied to interfaces. The match clause includes a count option to show hits for the corresponding matching conditions.

```
Core-A# show policy hitcounts remark_iSCSI_Initiator
Statistics for Policy remark_iSCSI_Initiator:

Interface 1/1/25 (in):
  Matched Packets  Configuration
10 class ip tcp_iscsi_initiator action pcp 4 action local-priority 4 action dscp
CS4
     6593750  10 match tcp any any eq iscsi count
20 class ipv6 v6_tcp_iscsi_initiator action pcp 4 action local-priority 4 action
dscp CS4
      0  10 match tcp any any eq iscsi count
```

Apply the policy to an interface to properly classify and mark the iSCSI traffic. The policy should be applied on the interface closest to the source originating the traffic, which is being remarked with the appropriate direction specified (in or out).

In the following configuration example, the policies are applied to the interface directly connected to the iSCSI Initiator and Target.

```
***Core A***

interface 1/1/25
  description Host_Initiator_iSCSI_Vlan10
  no shutdown
  mtu 9198
  flow-control priority rxtx 4
  no routing
  vlan trunk native 1
```

```
  vlan trunk allowed 10
  apply policy remark_iSCSI_Initiator in

interface 1/1/26
  description Target_iSCSI_Vlan10
  no shutdown
  mtu 9198
  flow-control priority rxtx 4
  no routing
  vlan trunk native 1
  vlan trunk allowed 10
  apply policy remark_iSCSI_Target in
```

Configuring QoS Trust and MTU

The Quality of Service (QoS) trust mode specifies how the switch assigns local priority values to ingress packets. The trust mode can be set globally for all interfaces, or individually for each interface. When the trust mode is set to CoS or DSCP, the switch translates the QoS settings in VLAN tags (for CoS), or the DS field in an IP header (for DSCP), to local priority values on the switch. The translation to local priority is controlled by the CoS map or DSCP map tables.

The following sample output shows the global trust mode set to DSCP and the corresponding mappings of CoS or DSCP values to local priority:

```
qos trust dscp
Core_A# show qos priority-servicing
  Trust Mode: DSCP
  Queue-Profile: lossless4p
                                                 Unicast   Multi-Destination
  PCP                                 Local-     Class of  Class of
  (CoS)      DSCP                     Priority   Service   Service
  ---------- ------------------------ ---------- --------- ---------
  1          8-15                     0          0         0
  0*         0-7                      1          0         0
  2          16-23                    2          0         0
  3          24-31                    3          0         0
  4          32-39                    4          1         1
  5          40-47                    5          0         0
  6          48-55                    6          2         2
  7          56-63                    7          2         2

*Default Service Class
```

When using a remark policy, assigning a QoS trust mode is optional.

It is a best practice to assign a trust mode appropriate for your security model. If the host and application are trusted, then a global qos trust policy can be applied. If applications or hosts are untrusted then it is recommended to NOT trust qos global and to create a policy for remarking the appropriate applications to its prioritized value.

The **show qos trust** command displays the current QoS global trust mode:

```
Core-A# show qos trust
qos trust dscp
```

The **qos trust dscp** command, used in the above example, assigns any received traffic with a DSCP marking to the associated local priority, including non-iSCSI traffic.

MTUs

In the switch configuration example below, there is a line configuring the MTU at 9198 bytes. The HPE ANW CX 8325 and 9300 Switch Series support jumbo frames, and the largest supported MTU size is 9198 bytes. MTU is configured at the physical port level and IP MTU is configured at the L3 Interface level (SVI/ROP).

- Jumbo frames are often recommended for iSCSI networks on which the connection is known to be low latency and dependable.
- Set the MTU values consistently across all switches and hosts utilizing the same ethernet fabric.
- To configure the ethernet MTU at the interface level, use the **mtu <size>** command. For L3 interface, use the **ip mtu <size>** command.

The following are some MTU configuration examples:

```
***Core A***
!
interface 1/1/25
 description Host_Initiator_iSCSI_Vlan10
 no shutdown
 mtu 9198
 flow-control priority rxtx 4
 no routing
 vlan trunk native 1
 vlan trunk allowed 10
 apply policy remark_iSCSI_Initiator in
!
interface vlan 10
 ip mtu 9198
 ip address 192.168.1.15/24
```

Configuring QoS Queue and Schedule Profile

When iSCSI is deployed in a network with no other sensitive data traffic, a three-queue profile model is recommended. This model creates 3 queues, where Queue 0 carries lossy traffic, Queue 1 carries iSCSI traffic, and Queue 2 carries signaling and protocol traffic.

The following example shows the three-queue profile model named **lossless4p** which is applicable for an iSCSI deployment, where a local priority of 4 (iSCSI) has been assigned to queue 1, which is considered as lossless traffic.

```
qos queue-profile lossless4p
 map queue 0 local-priority 0,1,2,3,5
 map queue 1 local-priority 4
 map queue 2 local-priority 6,7
```

To review the available qos queue-profiles, use the **show qos queue-profile** command. The following is the corresponding output for the **lossless4p** three-queue model:

```
Core-A# show qos queue-profile
profile_status profile_name
```

```
-------------- ------------
applied        factory-default
complete       lossless_qp4p

Core-A# show qos queue-profile lossless4p
queue_num local_priorities name
--------- ---------------- ----
0                0,1,2,3,5
1                4
2                6,7
```

The **factory-default** queue profile has a 1:1 mapping, as shown in the following example:

```
Core-A# show qos queue-profile factory-default
queue_num local_priorities name
--------- ---------------- ----
0     0          Scavenger_and_backup_data
1     1
2     2
3     3
4     4
5     5
6     6
7     7
```

A QoS schedule profile determines the order in which queues transmit a packet and the amount of service defined for each queue. A weighted schedule profile assigns relative servicing for each queue. Strict scheduling can be used to service queues purely on the basis of highest priority first (at the risk of starving lower-priority queues during high stress periods). A combination of strict and weighted scheduling offers more service to the highest priority queue when needed, while preserving scheduling between the remaining queues, thus decreasing the risk of starvation.

A QoS schedule profile is always applied on all interfaces, either from user configuration or using factory-default settings. Configuration can apply at the global level for all ports, as well as a specific schedule profile applied on a per-interface basis. If both global and interface-specific schedule profiles are configured, the interface specific configuration has precedence.

HPE ANW CX switches have a factory-default, which assigns Deficit Weighted Round Robin (DWRR) scheduling algorithms to all queues with an equal weight of 1. This schedule profile must be modified to achieve the lossless functionality for this solution.

The following is the lossless QoS schedule profile for iSCSI named **lossless_sp4p**:

```
qos schedule-profile lossless_sp4p
 dwrr queue 0 weight 20
 dwrr queue 1 weight 80
 strict queue 2
```

In the configuration above there are 3 different services for each queue profile. A weighted schedule profile is used for Queue 0 (Lossy Traffic) and for Queue 1 (Lossless Traffic), while Strict scheduling is used for Queue 2.

Queue 0 is set to DWRR with a weight of 20 for lossy traffic. Queue 1 is set to DWRR with a weight of 80 for iSCSI traffic. These weights provide an approximate 8 to 2 ratio for packets transmitted from Queue 1 versus Queue 0 when applied to an interface. This can be adjusted further based on application needs. The signaling traffic assigned to queue 2 is scheduled as strict scheduling, which means that packets in this queue are transmitted before all other queues.

To verify the currently available schedule profiles, run the following command:

```
Core-A# show qos schedule-profile
profile_status          profile_name
--------------          ------------
applied                 factory-default
complete                lossless_sp4p
complete                strict
```

The **show qos schedule-profile factory-default** command shows that **Algorithm** and **Weight** assigned to all the Queues is the same. Use the same command to verify and review the **lossless_sp4p** schedule profile with the corresponding algorithm and weights, as shown below:

```
Core-A# show qos schedule-profile factory-default
Queue                            Maximum     Bandwidth    Burst
Number  Algorithm      Weight    Bandwidth   Units        (KB)
------- -------------- -------   ----------  ----------   ----------
0        dwrr            1
1        dwrr            1
2        dwrr            1
3        dwrr            1
4        dwrr            1
5        dwrr            1
6        dwrr            1
7        dwrr            1

Core-A# show qos schedule-profile lossless_sp4p
queue_num algorithm     weight max-bandwidth_kbps burst_KB
--------- ------------- ------ ------------------ --------
0         dwrr             20
1         dwrr             80
2         strict
```

To apply queue profile and schedule profile globally to all ports use the command shown below. Check again the commands listed above to confirm that both profiles are shown as applied.

```
Core-A# apply qos queue-profile lossless4p schedule-profile lossless_sp4p

Core-A# show qos queue-profile
profile_status profile_name
-------------- ------------
complete       factory-default
applied        lossless_qp4p

Core-A# show qos schedule-profile
profile_status profile_name
-------------- ------------
complete       factory-default
applied        lossless_sp4p
complete       strict
```

Configuring QoS Pool

The Quality of Service (QoS) pool creates a packet buffer pool for lossless traffic by configuring lossless pool size, headroom buffer associated with the pool, and the priorities that are mapped to the pool.

The QoS pools on the HPE ANW CX 8325 and 9300 Switch Series are configurable and have the following attributes:

- The QoS pool option enables the creation of a packet buffer pool which is dedicated to lossless traffic.
- The QoS pool command allows the creation of a lossless pool size, headroom buffer associated with the pool, and the priorities that are mapped to that pool.
- The lossless pool size is a percentage of the total available buffer memory on the device.
- The headroom pool memory is allocated from the lossless pool and is used for storing packets that arrive at a port after a pause has been asserted.
- The sum of pool sizes cannot be larger than 90% of the available Kbyte memory.
- The headroom size cannot be greater than half of the lossless pool buffer size.

The command options are as follows:

```
switch(config)# qos pool <INDEX> lossless size <factory-default | PERCENT>
percent headroom <factory-default | KBYTES> kbytes priorities <PRIORITY 0-7>
```

The following lossless QoS pool is used in this example setup:

```
qos pool 1 lossless size 50.00 percent headroom 5000 kbytes priorities 4
```

Using the HPE ANW CX 8325-32C, CX 8325H-16C, CX 8325H-18Y4C Switches, the 50 percent above of the buffer means 16 MB of buffer space for traffic associated with local priority 4 with a headroom size of 5000 KB. For the case of the HPE ANW CX 9300 Switch Series, that 50 percent corresponds to 32 MB. The headroom pool memory is allocated from the lossless pool and is used for storing packets that arrive on a port after a pause has been asserted

.

HPE ANW CX 8325-32C, CX 8325H-16C, CX 8325H-18Y4C Switches

The HPE ANW CX 8325-32C, CX 8325H-16C, CX 8325H-18Y4C Switches have a total of 32 Mbytes of buffer space. The factory-default headroom size is 3072 KB (3 Mb).

The following is the corresponding output for **show qos pool**, showing 50% of the available buffer size to Lossless Pool 1 (11Mb), and 5000 KB of Headroom taken from Pool 1 size:

```
Core-A# show qos pool

Global Packet-Buffer Overview (in Kbytes)

                                      Sum of
                                       Port       Max
Packet Buffer Pools     Size          Limits     Usage
-------------------------------------------------------------
Lossy Pool              16004           --        10
Lossless Pool 1         11004 (50.00%)  2000      7937
Headroom Pool 1          5000           832       30
Reserved Memory           760           --        --
-------------------------------------------------------------
TOTAL:                   32768

Analysis:
```

```
Lossy Pool      was    0% utilized
Lossless Pool 1 was   72% utilized
Lossless Pool 1  is  450% over-provisioned
Headroom Pool 1 was    0% utilized
Headroom pool 1  is  500% over-provisioned
```

Run the **show qos pool statistics** command to check statistics.

```
Core-A # show qos pool statistics

Packet-Buffer Pool Statistics

Recent-Peak Interval: 283 seconds

Packet Buffer Pools   Total Size     Peak Use   Recent Peak  Current Use
-------------------   ----------   ----------   -----------  -----------
Lossy Pool                 16004           10             1            0
Lossless Pool 1            11004         7937             0            0
Headroom 1                  5000           30             0            0
Lossless Pool 2                0            0             0            0
Headroom 2                     0            0             0            0
Lossless Pool 3                0            0             0            0
Headroom 3                     0            0             0            0
```

Check the full configuration and validate ports distribution according to flow-control mode used the **show qos pool config** command . The output shows distribution of port with llfc configured:

```
Core-A# show qos pool config

Global Packet-Buffer Configuration and Status (in Kbytes)

                                  Sum of    Applied          Configured
                                   Port     Priority         Priority
Packet Buffer Pools  Size         Limits    Mapping          Mapping
-------------------- ------------ ---------- -------------  ---------------
Lossy Pool           16004            --    0,1,2,3,5,6,7   0,1,2,3,5,6,7
Lossless Pool 1      11004(50.00%)   2000    4               4
Headroom Pool 1      5000            1488    --              --
Reserved Memory      760               --    --              --
------------------------------------------------------------------------------
TOTAL:    32768

Pool-wise Per-Interface and Per-Flow-Control Mode Status

Lossy Pool

               Flow-Control
Interface      Mode            XOFF**  XON Delta  Headroom
-------------------------------------------------------------------------
1/1/1-1/1/24   none            --        --           --
1/1/27-1/1/32  none            --        --           --
-------------------------------------------------------------------------

Lossless Pool 1
               Flow-Control
Interface      Mode              XOFF**       XON Delta     Headroom
-------------- ------------ ----------------- -------------- ------------
```

```
1/1/25-1/1/26   llfc rxtx     Dynamic 1           12          328
-------------------------------------------------------------------------
TOTAL:                              --                      656

** Static XOFF limits are in Kbytes. Dynamic XOFF values are the alpha
factor used in the dynamic limit algorithm.
```

HPE ANW CX 9300S Switch

The HPE ANW CX 9300S Switch has a total of 84 Mbytes of buffer space. The factory-default headroom size is 3072 Kbytes (3 Mb).

The following is the corresponding output for **show qos pool**, showing 50% of the available buffer size to Lossless Pool 1 (36Mb), and 5000 KB of Headroom taken from Pool 1 size:

```
Core-A# show qos pool

Global Packet-Buffer Overview (in Kbytes)
                                Sum of
                                Port        Max
Packet Buffer Pools   Size      Limits      Usage
-------------------------------------------------------
Lossy Pool           40990          --         0
Lossless Pool 1      35990 (50.00%)  --        0
Headroom Pool 1      4999          816         0
Reserved Memory      1989           --        --
-------------------------------------------------------
TOTAL:               83968

Analysis:
 Lossy Pool      was    0% utilized
 Lossless Pool 1 was    63% utilized
 Headroom Pool 1 was    0% utilized
 Headroom pool 1  is  512% over-provisioned
```

Check statistics with the **show qos pool statistics** command.

```
Core-A # show qos pool statistics

Packet-Buffer Pool Statistics

Recent-Peak Interval: 0 seconds

Packet Buffer Pools   Total Size  Peak Use   Recent Peak  Current Use
------------------    ----------  ---------- -----------  -----------
Lossy Pool             40990      0           0             0
Lossless Pool 1        35990      8397        0             0
Headroom 1             4999       32          0             0
Lossless Pool 2        0          0           0             0
Headroom 2             0          0           0             0
Lossless Pool 3        0          0           0             0
Headroom 3             0          0           0             0

All buffer sizes are in Kbytes.
```

Check the full configuration and validate ports distribution according to flow-control mode using **show qos pool config**. The output shows distribution of ports with pfc configured:

```
Core-A# show qos pool config


Global Packet-Buffer Configuration and Status (in Kbytes)
                                  Sum of    Applied       Configured
                                  Port      Priority      Priority
Packet Buffer Pools   Size        Limits    Mapping       Mapping
-------------------------------------------------------------------------------
Lossy Pool            40990         --      0,1,2,3,5,6,7  0,1,2,3,5,6,7
Lossless Pool 1       35990 (50.00%)  --      4              4
Headroom Pool 1       4999          1632    --             --
Reserved Memory       1989          --      --             --
-------------------------------------------------------------------------------
TOTAL:                        83968

Pool-wise Per-Interface and Per-Flow-Control Mode Status

Lossy Pool
               Flow-Control
Interface      Mode            XOFF**         XON Delta  Headroom
-------------------------------------------------------------------
1/1/1-1/1/24   none            --                --        --
1/1/27-1/1/40  none            --                --        --
-------------------------------------------------------------------

Lossless Pool 1

               Flow-Control
Interface      Mode                  XOFF**       XON Delta  Headroom
-------------------------------------------------------------------
1/1/25-1/1/26   priority 4 rxtx      Dynamic 1        12       408
-------------------------------------------------------------------
TOTAL:          --                                            816

** Static XOFF limits are in Kbytes. Dynamic XOFF values are the alpha
factor used in the dynamic limit algorithm.
```

HPE ANW CX 9300-32D Switch

The HPE ANW 9300-32D Switch has a total of 132 Mb of packet-buffer memory divided evenly between two Traffic Managers (TMs), giving each TM 66 Mbytes. The ports are divided between the two TMs, allowing 66 Mbytes of packet-buffer memory to be shared by all the ports assigned to a given TM. By default, all 66 Mbytes of each TM's packet-buffer memory are in one pool (the Lossy pool). The factory-default headroom size is 3072 KB (3 MB).

The distribution of ports and the TMs is as follows:

- TM1

  Port 1/1/9 to 1/1/12 and Port 1/1/21 to 1/1/32

  Total= 16 MB
- TM2

  Port 1/1/1 to 1/1/8 and Port 1/1/13 to 1/1/20

  Total= 16 MB

The **show qos pool config** command provides information about the pool assignation by port and the corresponding settings of flow control.

NOTE

The ports associated with TM1 are shown with the corresponding Pool. Ports that have link level flow control configured, like ports 1, 2, 25, and 26, are bundled depending on the use and the seetings of XOFF and XON values.

The following is the sample output of the **show qos pool config** command:

```
Core-A# show qos pool config

Global Packet-Buffer Configuration and Status (in Kbytes)
Traffic Manager 1 Configuration

                      Sum of        Applied    Configured
                      Port          Priority   Priority
Packet Buffer Pools   Size          Limits     Mapping         Mapping
-------------------------------------------------------------------------------
Lossy Pool            32355              --     0,1,2,3,5,6,7   0,1,2,3,5,6,7
Lossless Pool 1       27355 (50.00%)     --               4               4
Headroom Pool 1       4999              652              --              --
Reserved Memory       2875               --              --              --
-------------------------------------------------------------------------------
TOTAL:                    67584

Pool-wise Per-Interface and Per-Flow-Control Mode Status

Lossy Pool

                Flow-Control
Interface             Mode             XOFF**         XON Delta  Headroom
-------------------------------------------------------------------------
1/1/9-1/1/12          none             --                    --        --
1/1/21-1/1/24         none             --                    --        --
1/1/27-1/1/32         none             --                    --        --
-------------------------------------------------------------------------

Lossless Pool 1

                Flow-Control
Interface             Mode             XOFF**         XON Delta  Headroom
-------------------------------------------------------------------------
1/1/25-1/1/26         llfc rxtx        Dynamic 1             12       326
-------------------------------------------------------------------------
TOTAL:                                 --                            652

** Static XOFF limits are in Kbytes. Dynamic XOFF values are the alpha
factor used in the dynamic limit algorithm.

Traffic Manager 2 Configuration

                    Sum of        Applied     Configured
                    Port          Priority    Priority
Packet Buffer Pools Size          Limits      Mapping         Mapping
-------------------------------------------------------------------------------
Lossy Pool          32355              --     0,1,2,3,5,6,7   0,1,2,3,5,6,7
Lossless Pool 1     27355 (50.00%)     --               4               4
 Headroom Pool 1     4999              652              --              --
Reserved Memory      2875              --               --              --
-------------------------------------------------------------------------------
TOTAL:                    67584
```

```
Pool-wise Per-Interface and Per-Flow-Control Mode Status

Lossy Pool

                Flow-Control
Interface             Mode             XOFF**          XON Delta  Headroom
-------------------------------------------------------------------------
1/1/3-1/1/8           none             --                    --        --
1/1/13-1/1/20         none             --                    --        --
-------------------------------------------------------------------------
```

The **show qos pool** and **show qos pool statistics** CLI commands displays the QoS pool usage and statistics. The following is the corresponding output for the lossless QoS pool, showing 50% of the available buffer size to Pool 1, and the 5000 KB of Headroom taken from Pool 1 size. Check the statistics with the **show qos pool statistics** command.

```
Core-A# show qos pool

Global Packet-Buffer Overview (in Kbytes)
Traffic Manager 1 Overview

                                          Sum of
                                          Port        Max
Packet Buffer Pools       Size            Limits      Usage
-----------------------------------------------------------
Lossy Pool               32355                --          1
Lossless Pool 1          27355 (50.00%)       --          4
  Headroom Pool 1         4999               326          0
Reserved Memory           2875                --         --
-----------------------------------------------------------
TOTAL:                   67584

Analysis:
 Lossy Pool      was    0% utilized
 Lossless Pool 1 was    0% utilized
 Headroom Pool 1 was    0% utilized
 Headroom pool 1  is 1433% over-provisioned

Traffic Manager 2 Overview

                                          Sum of
                                          Port        Max
Packet Buffer Pools       Size            Limits      Usage
-----------------------------------------------------------
Lossy Pool                32355                --       1109
Lossless Pool 1           27355 (50.00%)       --          0
 Headroom Pool 1           4999                978          0
Reserved Memory            2875                 --         --
-----------------------------------------------------------
TOTAL:                   67584

Analysis:
 Lossy Pool      was    3% utilized
 Lossless Pool 1 was    0% utilized
 Headroom Pool 1 was    0% utilized
 Headroom pool 1  is  411% over-provisioned

Core-A# show qos pool statistics
```

```
Packet-Buffer Pool Statistics

Traffic Manager 1 Statistics

Recent-Peak Interval: 0 seconds

Packet Buffer Pools   Total Size     Peak Use   Recent Peak  Current Use
-------------------   ----------   ----------   -----------  -----------
Lossy Pool                 32355        25890             0            0
Lossless Pool 1            27355            5             0            0
 Headroom 1                 4999            0             0            0
Lossless Pool 2                0            0             0            0
 Headroom 2                    0            0             0            0
Lossless Pool 3                0            0             0            0
 Headroom 3                    0            0             0            0

Traffic Manager 2 Statistics

Recent-Peak Interval: 0 seconds

Packet Buffer Pools   Total Size     Peak Use   Recent Peak  Current Use
-------------------   ----------   ----------   -----------  -----------
Lossy Pool                 32355         1015             1            0
Lossless Pool 1            27355          766             0            0
 Headroom 1                 4999            0             0            0
Lossless Pool 2                0            0             0            0
 Headroom 2                    0            0             0            0
Lossless Pool 3                0            0             0            0
 Headroom 3                    0            0             0            0

All buffer sizes are in Kbytes.
```

Configuring LLFC

The Link Level Flow Control (LLFC) is the standard mechanism for temporarily pausing the transmission of data on Ethernet networks. The goal of this mechanism is to avoid packet loss in the presence of network congestion. This allows loss-sensitive protocols, such as iSCSI, to coexist with traditional loss-tolerant protocols over the same unified fabric.

For the HPE ANW CX 8325 and 9300 Switch Series, a pool is needed as described in the Configuring QoS Pool section. The override-negotiation configuration option is necessary for hosts and arrays that do not support LLFC configuration via auto-negotiation functionality.

In the configuration example below, the Initiator and Target interfaces have been configured with Link Level Flow Control. The **rxtx** option, in the configuration below, enables to receive and transmit pause frames. Pool 1 is the associated lossless pool that serves as a buffer for paused traffic.

```
***CoreA***
interface 1/1/25
 description Host_Initiator_iSCSI_Vlan10
 no shutdown
 mtu 9198
 flow-control rxtx pool 1 override-negotiation
 no routing
 vlan trunk native 1
 vlan trunk allowed 10
 apply policy remark_iSCSI_Initiator in
```

```
interface 1/1/26
 description Target_iSCSI_Vlan10
 no shutdown
 mtu 9198
 flow-control rxtx pool 1 override-negotiation
 no routing
 vlan trunk native 1
 vlan trunk allowed 10
 apply policy remark_iSCSI_Target in
```

Run the following command to confirm that flow control is configured and enabled:

```
Core-A# show int 1/1/25 flow-control

Flow Control Watchdog Settings
 Trigger Timeout:  100 milliseconds
 Resume Time:      100 milliseconds
----------- -------------------- ------------- ----------
Port            Flow                Watchdog    Watchdog
                Control             Status      Timeouts
----------- -------------------- ------------- ----------
1/1/25      config: llfc rxtx
            status: llfc rxtx
```

Run the following command to confirm that flow control is configured and enabled:

```
Core-A# show int 1/1/26 flow-control detail
Interface 1/1/26 is up
 Admin state is up
 Link state: up for 4 days (since Wed May 22 19:01:25 UTC 2024)
 Flow-control: llfc rxtx
 Flow-control watchdog: disabled

 Statistics                    RX                   TX
 --------------------- -------------------- --------------------
 Dot3 Pause Frames             0                    0
```

Figure 1 shows the iSCSI Initiator connected to port 25 in both switches. To avoid data loss from congestion, the switch uses the previously configured QoS features to provide lossless traffic flow.

The following output shows the switch sending pause frames to the Initiator to ensure iSCSI packets are not dropped when buffer resources are low.

```
Core-A# show int 1/1/25 flow-control detail
Interface 1/1/25 is up
 Admin state is up
 Link state: up for 1 day (since Wed May 22 18:54:12 UTC 2024)
 Flow-control: llfc rxtx
 Flow-control watchdog: disabled

 Statistics                    RX                   TX
 --------------------- -------------------- --------------------
 Dot3 Pause Frames              0                   2140
```

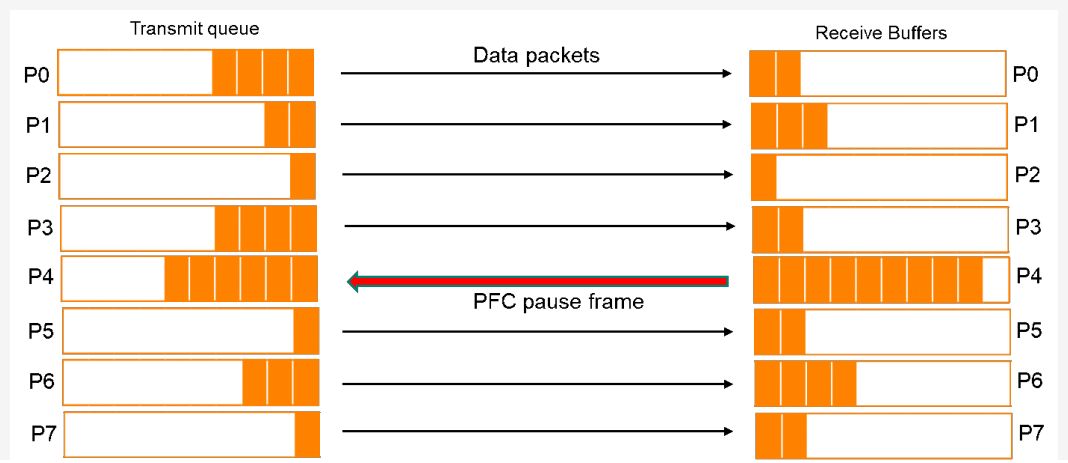
Configuring PFC

The Priority-based flow control (PFC) provides an enhancement to the legacy Ethernet flow control (LLFC) standard. PFC operates at Layer 2 and supports Layer 2 and Layer 3 network connectivity. The standard Ethernet flow control pauses all traffic on an Ethernet link when the input interface buffer is full, impacting all traffic flows.

The 802.1Qbb standard for PFC allows a device to separately suspend up to eight different traffic classes. AOS-CX switching devices support a variable number of PFC-enabled priorities on each port, depending on the model.

It is recommended to only assign a single PFC priority per queue and avoid combining more than one PFC priority into a single queue, because pausing any one of the priorities results in both being paused by the switch.

A port configured with PFC sends a pause frame (XOFF) to the transmitting station when the buffer becomes full for a specific traffic class without impacting other traffic classes. The pause frame signals the transmitting station to stop sending packets of that traffic class but allows all other traffic to continue.



The HPE ANW CX 9300 Switch Series supports 8 queues per interface but a maximum of 7 queues can be configured for PFC.

The HPE ANW CX 8325 Switch Series supports 8 queues per interface but a maximum of 3 queues can be configured for PFC.

NOTE

All the traffic for a given queue will be considered lossless and paused upon receiving a pause frame. For example, if two applications have different DSCP values but map to the same local priority and that queue for that local priority is paused, then both applications will be paused.

In the below configuration, PFC has been enabled with RXTX (Receive and Transmit Pause Frames) with priority 4.

In this configuration the switch transmits a Pause frame when congestion occurs and pauses transmission when a flow control frame is received from a host. This only occurs for traffic that is received with local priority 4. In this configuration the switch transmits a Pause frame when the congestion occurs and pauses transmission when a flow control frame is received from a host. This only occurs for traffic that is received with local priority 4, which means that only pause frames are transmitted to the queue using P4. This is how you stop iSCSI traffic and prevent losses during congestion.

NOTE

PFC is enabled at the physical interface level.

```
***Core A***
interface 1/1/25
 description Host_Initiator_iSCSI_Vlan10
 no shutdown
 mtu 9198
 flow-control priority rxtx 4
 no routing
 vlan trunk native 1
 vlan trunk allowed 10
 apply policy remark_iSCSI_Initiator in
…
…
***Core B***
interface 1/1/25
 description Host_Initiator_Vlan_20
 mtu 9198
 flow-control priority rxtx 4
 no routing
 vlan trunk native 1
 vlan trunk allowed 20
 apply policy remark_iSCSI_Initiator in
```

Use the **show int 1/1/X flow-control** command to confirm that priority flow control is configured and enabled:

```
Core-A# show int 1/1/25 flow-control

Flow Control Watchdog Settings
Trigger Timeout:  100 milliseconds
Resume Time:      100 milliseconds

----------- ------------- ------------- --------
Port          Flow        Watchdog      Watchdog
              Control     Status        Timeouts
----------- ------------- ------------- --------

1/1/25      config: pfc rxtx-4
            status: pfc rxtx-4
```

Use the **show int 1/1/X flow-control detail** to confirm that pause frames have been forwarded only for Priority 4:

```
Core-A# show int 1/1/25 flow-control detail
Interface 1/1/25 is up
Admin state is up
Link state: up for 6 days (since Sun Mar 11 14:43:36 UTC 2024)
Flow-control: pfc rxtx-4
Flow-control watchdog: disabled

Statistics                RX                    TX
------------------- ---------------- ------------------
Priority 0 Pauses         0                     0
Priority 1 Pauses         0                     0
Priority 2 Pauses         0                     0
Priority 3 Pauses         0                     0
Priority 4 Pauses         0                 18402
Priority 5 Pauses         0                     0
```

```
Priority 6 Pauses          0                    0
Priority 7 Pauses          0                    0
Total Pause Frames         0                18402
```

Limitation

Due to an ASIC limitation, traffic with lossless pool assignment is based on PCP value. Traffic with a PCP value matching a local priority for lossy traffic is put in the lossy pool, not the lossless pool. To avoid this, remark PCP value to 4 in the Policy, as shown in the following configuration example. This ensures that the ingress traffic is linked to the corresponding local-priority attached with the lossless queue.

The QoS Classes and Policies explained in the Configuring QoS Classes and Policies provides a solution:

```
***Setup Applies for both, Core A and Core B***
class ip tcp_iscsi_initiator
 10 match tcp any any eq iscsi count
…
class ipv6 v6_tcp_iscsi_initiator
 10 match tcp any any eq iscsi count
…
policy remark_iSCSI_Initiator
 10 class ip tcp_iscsi_initiator action pcp 4 action local-priority 4 action dscp
CS4
 20 class ipv6 v6_tcp_iscsi_initiator action pcp 4 action local-priority 4 action
dscp CS4
```

Alternatively, you can match packets tagged with DSCP CS4 value:

```
***Setup Applies for both, Core A and Core B***

class ip cs4
 10 match any any any dscp CS4 count
policy remark_local4
 10 class ip cs4 action local-priority 4
interface lag 128
 apply policy remark_local4 in
```

Check the host side to enable PFC manually, as explained in the Configuring PFC on Nvidia (Mellanox) NIC Adapters section to complete the PFC setup.

Configuring PFC with DCBX

The Data Center Bridging Exchange protocol (DCBx) is a discovery and capability exchange protocol for communicating The Data Center Bridging configuration information between link peers. DCBx is specified as part of IEEE 802.1Qaz-2011 standard. DCBx uses LLDP as the underlying protocol for the exchange of parameters with the peer. The DCBx parameters are exchanged as LLDP TLVs.

The DCBx is an extension of LLDP used by DCB devices to exchange configuration information with directly connected peers. Its applications include:

- Discovery of peer capabilities.
- Detection of misconfigurations and identifying configuration mismatches.
- Advising a DCB peer regarding the DCB configurations it should adopt.

DCBx guidelines

- DCBx is disabled by default. DCBx can be enabled globally or within a specific interface.
- LLDP must be enabled on the interfaces supporting DCBx.
- DCBx should be configured on all L2 interfaces.
- L3 hops which leverage PFC and ECN do not require DCBx, however, it is recommended to enable it.
- DCBx is only supported on physical interfaces and not on management or logical interfaces, similar to how LLDP behaves.
- DCBx is not essential between host and switch but is highly desirable to avoid manual PFC configurations of host devices.
- AOS-CX Switches supports:
  - DCBx Converged Enhanced Ethernet (CEE)
  - DCBx IEEE 802.1Qaz
- The IEEE DCBx IEEE 802.1Qaz is the default version when DCBx is enabled.
- AOS-CX advertises DCBx with the willing bit set to 0 in all TLVs. This tells the peer that the switch is not willing to change its configuration to match the peer's configuration.
- When a peer device does not support the configured DCBx version (IEEE or CEE), a misconfiguration error is displayed in the **show dcbx interface** output.

A DCBx-enabled switch can request its peer to map certain application traffic to a priority using the DCBx TLV subtype (0x0c). Whether the host accepts the QoS parameters or rejects them is dependent on the host's local DCBx NIC willing "W" bit state:

- If set to 1—Indicates that the adapter is willing to accept QoS parameters from the remote peer.
- If set to 0—Operational QoS parameters are always resolved from the local Host NIC QoS parameters.

The DCBx is expected to operate over a point-to-point link. If multiple LLDP neighbors are detected, then.

The DCBx behaves if its peer DCBx TLVs are not present and behaves this way until the multiple LLDP neighbor condition is no longer present.

The LLDP must be enabled globally before any DCBx configuration. The LLDP is enabled by default, but if necessary, can be enabled with the **lldp enable** command.

The following examples describe how to enable DCBx globally or at the interface level and how to specify either the IEEE or CEE version:

- Enabling DCBx globally (IEEE Version):

```
switch(config)# lldp dcbx
```

- Enabling the CEE version of DCBx globally:

```
switch(config)# lldp dcbx version cee
```

- Enabling DCBx for an interface (IEEE version):

```
switch(config-if)# lldp dcbx
```

- Enabling the CEE version of DCBx for an interface:

```
switch(config-if)# lldp dcbx version cee
```

In the Final iSCSI with LLFC and PFC Configuration on Air Gap Core Topology section, Core switches use **lldp dcbx** to enable IEEE Version. In this tested scenario, DCBx TLV advertisement is used by switch to advertise to the host to use PCP value of 4 for iSCSI traffic and places it in queue 1 (Lossless queue) with command **dcbx application iscsi priority 4**.

Not all drivers support this feature so additional setup could be necessary at host side to forward iSCSI traffic with PCP value equal to 4 or DSCP value that matches a local priority associated with lossless queue.

Below is the output of show dcbx interface at port 25, connected to host initiator. The information displayed shows that initiator have dcbx enable, both are using the same dcbx version and initiator is willing to receive the setup of the switch. Also, note that Priority 4 is shown as True, because that interface is configured with PFC priority 4.

```
Core-A# show dcbx int 1/1/25
DCBx admin state           : enabled
DCBx operational state     : active
DCBx version               : local = ieee, remote = ieee
PFC operational state      : active

Mismatch  Advertisement              Local    Peer
--------- -------------------------- -------- ----
Priority Flow Control (PFC)
 -> Willing:                         No       Yes
 MACsec Bypass Capability:  No          No
 -> Max PFC Traffic Classes:     7           8
 Priority 0:                         False    False
 Priority 1:                                  False    False
 Priority 2:                         False    False
 Priority 3:                         False    False
 Priority 4:                         True     True
 Priority 5:                         False    False
 Priority 6:                         False    False
 Priority 7:                                  False    False

Enhanced Transmission Selection (ETS)
 -> Willing:                         No       Yes
 Credit-Based Shaper:                No       No
 -> Max Traffic Classes:             3        8
 Priority 0:                         Class 0  Class 0
 Priority 1:                         Class 0  Class 0
 Priority 2:                         Class 0  Class 0
 Priority 3:                         Class 0  Class 0
 Priority 4:                         Class 1  Class 1
 Priority 5:                         Class 0  Class 0
 Priority 6:                         Class 2  Class 2
 Priority 7:                         Class 2  Class 2
 Class 0:                            ETS 20   ETS 20
 Class 1:                            ETS 80   ETS 80
 Class 2:                            Strict   Strict
 Class 3:                            ETS 0    ETS 0
 Class 4:                            ETS 0    ETS 0
 Class 5:                            ETS 0    ETS 0
```

```
 Class 6:                          ETS 0    ETS 0
 Class 7:                          ETS 0    ETS 0

Application Priority Map:
Mismatch  Protocol     Protocol ID     Local        Peer
                       Priority     Priority
--------- ------------ --------------- ------------ -------------
-> iscsi                         4
```

See the AOS-CX Switch Documentation Portal, select switch model to look for the latest *Quality of Service* guide, and also CLI Configuration Guide for more details and configuration options for all protocols mentioned in the previous sections.

Deploying Ethernet Smart NICs for iSCSI

Ethernet Network Interface Cards (NICs) are suited for network connectivity for high-performance computing, secure data center connectivity, and AI/ML applications in cloud-scale and enterprise environments.

The iSCSI Initiator, uses an iSCSI driver to enable target resource recognition and attachment to the iSCSI Target over IP. Various adapters can be used, such as standard NICs, iSCSI HBA cards, converged network adapters, and Smart NICs.

The following table provides information of ethernet adapters validated with this solution:

Table 2: *Validated Adapter*

Part Number	ASIC	Ports	Form Factor	Bus Type
P25960-B21	HPE **Mellanox** MCX623106AS-CDAT Ethernet 100Gb Dual Port QSFP56	2x 100G	Stand Up	PCIe

This chapter includes the following topics:

Installing and Configuring Nvidia (Mellanox) Ethernet Smart NIC on Red Hat

To install and configure Nvidia (Mellanox) NIC Adapters on Red Hat, complete the following steps:

Prerequisite—You must complete the Red Hat Enterprise host set up, confirm the subscription, and run the latest upgrades.

1. Use the following command to check the installed and available kernel updates:

```
[root@storage]# dnf list kernel
```

2. Use the following command for a full system update:

```
[root@storage]# dnf update -y --quiet
```

NOTE

Use the quiet option.

3. Use the following commands to review the current NIC part description and firmware version:

```
root@storage]# ethtool -i ens3f0np0
driver: mlx5_core
version: 5.8-3.0.7
firmware-version: 22.35.1012 (MT_0000000437)
expansion-rom-version:
bus-info: 0000:d8:00.0
supports-statistics: yes
supports-test: yes
supports-eeprom-access: no
supports-register-dump: no
supports-priv-flags: yes
```

```
[root@storage]# mstflint -d d8:00.0 q
Image type:            FS4
FW Version:            22.35.1012
FW Release Date:       28.10.2022
Product Version:       22.35.1012
Rom Info:              type=UEFI version=14.28.15 cpu=AMD64,AARCH64
type=PXE version=3.6.804 cpu=AMD64
Description:           UID                GuidsNumber
Base GUID:             88e9a4ffffcaa2ae        4
Base MAC:              88e9a4caa2ae            4
Image VSD:             N/A
Device VSD:            N/A
PSID:                  MT_0000000437
```

This upgrades the firmware version of the Mellanox NIC.

For more information, see Firmware Update Instructions.

Download the Mellanox firmware tools and the firmware that correspond to the model and operating system.

For more information, refer to the following:

NOTE

- Software and firmware upgrades—How to update Red Hat Enterprise Linux through minor releases and Extended Update Support and Downloading Mellanox OFED - MLNX_OFED v4.6-1.0.1.1- NVIDIA Networking Docs for Mellanox OFED Tools.
- Mellanox QoS tools—Nvidia Support

4. Use the following command to unzip the Firmware file downloaded to server:

```
[root@storage]# unzip fw-ConnectX6Dx-rel-22_35_3502-MCX623106AS-CDA_Ax-UEFI-
14.29.15-FlexBoot-3.6.902.signed.bin.zip
Archive:  fw-ConnectX6Dx-rel-22_35_3502-MCX623106AS-CDA_Ax-UEFI-14.29.15-
FlexBoot-3.6.902.signed.bin.zip
inflating: fw-ConnectX6Dx-rel-22_35_3502-MCX623106AS-CDA_Ax-UEFI-14.29.15-
FlexBoot-3.6.902.signed.bin
[root@storage]#
[root@storage]# ls | grep fw-ConnectX6D
fw-ConnectX6Dx-rel-22_35_3502-MCX623106AS-CDA_Ax-UEFI-14.29.15-FlexBoot-
3.6.902.signed.bin
fw-ConnectX6Dx-rel-22_35_3502-MCX623106AS-CDA_Ax-UEFI-14.29.15-FlexBoot-
3.6.902.signed.bin.zip
```

Complete this step in all the initiator hosts and targets that uses this NVIDIA (Mellanox) card.

5. Use the following command to confirm whether firmware is updated properly:

```
[root@storage]# flint -d /dev/mst/mt4125_pciconf0 -i fw-ConnectX6Dx-rel-22_35_
3502-MCX623106AS-CDA_Ax-UEFI-14.29.15-FlexBoot-3.6.902.signed.bin b
Current FW version on flash:  22.35.1012
New FW version:               22.35.3502
FSMST_INITIALIZE -   OK
Writing Boot image component -   OK
-I- To load new FW run mlxfwreset or reboot machine.
[root@storage]#
[root@storage]# mlxfwreset -d /dev/mst/mt4125_pciconf0 reset
Minimal reset level for device, /dev/mst/mt4125_pciconf0:
3: Driver restart and PCI reset
Continue with reset?[y/N] y
-I- Sending Reset Command To Fw             -Done
-I- Stopping Driver                         -Done
-I- Resetting PCI                           -Done
-I- Starting Driver                         -Done
-I- Restarting MST                          -Done
-I- FW was loaded successfully.
```

```
[root@storage]# mstflint -d d8:00.0 q
Image type:            FS4
FW Version:            22.35.3502
FW Release Date:       27.12.2023
Product Version:       22.35.3502
Rom Info:              type=UEFI version=14.29.15 cpu=AMD64,AARCH64
type=PXE version=3.6.902 cpu=AMD64
Description:           UID                GuidsNumber
Base GUID:             88e9a4ffffcaa2ae        4
Base MAC:              88e9a4caa2ae            4
Image VSD:             N/A
Device VSD:            N/A
PSID:                  MT_0000000437
Security Attributes:   secure-fw
```

Configuring iSCSI Initiator on Nvidia (Mellanox) Ethernet Smart NIC on Red Hat

To configure and enable iSCSI Initiator in RHEL, complete the following steps:

1. Use the following commands to start the iSCSI service demon and confirm that it is currently running:

```
[root@storage]#systemctl start iscsid.service
[root@storage]#systemctl enable iscsid.service
```

```
[root@storage]#systemctl status iscsid.service
● iscsid.service - Open-iSCSI
Loaded: loaded (/usr/lib/systemd/system/iscsid.service; enabled; preset:
disabled)
Active: active (running) since Wed 2024-01-17 18:16:33 CST; 3 weeks 6 days ago
TriggeredBy: ● iscsid.socket
Docs: man:iscsid(8)
man:iscsiuio(8)
man:iscsiadm(8)
Main PID: 2681 (iscsid)
Status: "Ready to process requests"
Tasks: 1 (limit: 1231947)
Memory: 13.9M
CPU: 8.083s
CGroup: /system.slice/iscsid.service
└─2681 /usr/sbin/iscsid -f
```

2. Use the **yum install iscsi-initiator-utils** command to install iSCSI Initiator utilities on the host:

```
[root@storage]#yum install iscsi-initiator-utils
```

3. Use the **cat etc/iscsi/initiatorname.iscsi** command to check the Initiator name. The IQN (iSCSI Qualified Name) name is to be added as a valid Initiator in the Target. When an Initiator group is created, an Access Control List (ACL) is attached using the IQN names. Change the Initiator name with any text editor if that was changed in the target ACL:

```
[root@storage]# cat /etc/iscsi/initiatorname.iscsi
InitiatorName=iqn.1994-05.com.redhat:6beedce447e3
```

4. Use the following commands to initiate discover and login process to the target using the displayed target IQN. The host attempts to register against a target (Target Host hosted in the Linux device) associated with discovery IP 192.168.1.10 for VLAN 10 and IP 192.168.2.10 for VLAN 20. These IPs are configured in two different NICs and represent the two options to reach the volume configured in that target.

```
[root@storage]# iscsiadm -m discovery -t st -p 192.168.1.10
192.168.1.10:3260,1 iqn.2003-01.org.linux-iscsi.dl360_
target.x8664:sn.04eaf6dd4e42
[root@ storage]# iscsiadm -m discovery -t st -p 192.168.2.10
192.168.2.10:3260,1 iqn.2003-01.org.linux-iscsi.dl360_
target.x8664:sn.04eaf6dd4e42
[root@storage]# iscsiadm -m node -T iqn.2003-01.org.linux-iscsi.dl360_
target.x8664:sn.04eaf6dd4e42 -l
```

```
Logging in to [iface:
default, target: iqn.2003-01.org.linux-iscsi.dl360_
target.x8664:sn.04eaf6dd4e42,
portal: 192.168.1.10,3260]
Logging in to [iface:
default, target: iqn.2003-01.org.linux-iscsi.dl360_
target.x8664:sn.04eaf6dd4e42,
portal: 192.168.2.10,3260]
Login to [iface: default,
target: iqn.2003-01.org.linux-iscsi.dl360_target.x8664:sn.04eaf6dd4e42,
portal:
192.168.1.10,3260] successful.
Login to [iface: default,
target: iqn.2003-01.org.linux-iscsi.dl360_target.x8664:sn.04eaf6dd4e42,
portal:
192.168.2.10,3260] successful.
```

IQN is the same for both target IPs, related to same volume enable for iSCSI.

5. Use the following commands to find the iSCSI disk name and create a file system on it.

```
[root@storage]# grep "Attached SCSI" /var/log/messages
Jun 21 12:45:16 storage kernel: sd 3:0:0:0  [sdc] Attached SCSI disk.
```

6. Use the **lsblk** command to see the new disk attached to the local drives.

```
[root@storage]# lsblk
NAME                    MAJ:MIN RM  SIZE RO TYPE MOUNTPOINTS
sda                       8:0    0  2.2T  0 disk
├─sda1                    8:1    0  600M  0 part /boot/efi
├─sda2                    8:2    0    1G  0 part /boot
├─sda3                    8:3    0 55.1G  0 part
│ ├─rhel-pool00_tmeta   253:0    0   40M  0 lvm
│ │ └─rhel-pool00-tpool 253:2    0   40G  0 lvm
│ │   ├─rhel-root       253:3    0   16G  0 lvm  /
│ │   ├─rhel-pool00     253:5    0   40G  1 lvm
│ │   └─rhel-home       253:6    0   24G  0 lvm  /home
│ ├─rhel-pool00_tdata   253:1    0   40G  0 lvm
│ │ └─rhel-pool00-tpool 253:2    0   40G  0 lvm
│ │   ├─rhel-root       253:3    0   16G  0 lvm  /
│ │   ├─rhel-pool00     253:5    0   40G  1 lvm
│ │   └─rhel-home       253:6    0   24G  0 lvm  /home
│ └─rhel-swap           253:4    0    4G  0 lvm  [SWAP]
sdb                      8:16    1    0B  0 disk
sdc                      8:48    0  36.5G  0 disk

[root@storage]# mkfs.ext4 /dev/sdc
```

7. Use the following commands to mount the disk to the file system. Please refer to Configuring Device Mapper Multipath and Enabling LLFC on Red Hat for additional information.

```
mkdir /mount/point
mount /dev/disk_name /mount/point
```

```
[root@storage /]# mkdir /Target_f
[root@storage]# find Target*
Target_f
[root@storage]# mount /dev/sdc /Target_f
```

8. Use the following commands to edit the **/etc/fstab** file to mount the file system automatically when the system boots. Use any available text editor command like **vi** or **nano** to upgrade the fstab file.

```
[root@storage]# vi /etc/fstab
[root@storage/] nano /etc/fstab
[root@storage]# /dev/disk_name /mount/point ext4_netdev 0 0
```

```
[root@storage/] nano /etc/fstab
#
# /etc/fstab
# Created by anaconda on Wed Jun  7 19:47:53 2023
#
# Accessible filesystems, by reference, are maintained under '/dev/disk/'.
# See man pages fstab(5), findfs(8), mount(8) and/or blkid(8) for more info.
#
# After editing this file, run 'systemctl daemon-reload' to update systemd
# units generated from this file.
#
/dev/mapper/rhel-root   /                       xfs     defaults        0 0
UUID=04196ed3-c195-4e19-884f-082d89539e65 /boot xfs     defaults        0 0
UUID=EE5D-EF5D          /boot/efi               vfat
umask=0077,shortname=winnt 0 2
/dev/mapper/rhel-home   /home                   xfs     defaults        0 0
/dev/mapper/rhel-swap   none                    swap    defaults        0 0
/dev/sdc /Target_f ext4_netdev 0 0
```

9. Use the following command to confirm whether the new disk (external target) was properly mounted to a local folder:

```
[root@storage /]# lsblk
NAME                    MAJ:MIN RM  SIZE RO TYPE MOUNTPOINTS
sda                       8:0    0  2.2T  0 disk
├─sda1                    8:1    0  600M  0 part /boot/efi
├─sda2                    8:2    0    1G  0 part /boot
├─sda3                    8:3    0 55.1G  0 part
│ ├─rhel-pool00_tmeta   253:0    0   40M  0 lvm
│ │ └─rhel-pool00-tpool 253:2    0   40G  0 lvm
│ │   ├─rhel-root       253:3    0   16G  0 lvm  /
│ │   ├─rhel-pool00     253:5    0   40G  1 lvm
│ │   └─rhel-home       253:6    0   24G  0 lvm  /home
│ ├─rhel-pool00_tdata   253:1    0   40G  0 lvm
```

```
│ │ └─rhel-pool00-tpool 253:2    0   40G  0 lvm
│ │   ├─rhel-root       253:3    0   16G  0 lvm  /
│ │   ├─rhel-pool00     253:5    0   40G  1 lvm
│ │   └─rhel-home       253:6    0   24G  0 lvm  /home
│ └─rhel-swap           253:4    0    4G  0 lvm  [SWAP]
└─sda4                    8:4    0 36.5G  0 part
sdb                       8:16   1    0B  0 disk
sdc                       8:48   0   36G  0 disk /Target_f
```

10. Transfer or copy files to folder and check the target management console to confirm files transfer.

Configuring Device Mapper Multipath and Enabling LLFC on Red Hat

The Device Mapper (DM) multipath can provide failover in an active-passive or active-active configuration. In an active-passive configuration, only a subset of the paths is used at any time for Input-Output (I/O) operations. If any element of an I/O path such as the cable, switch, or gateway fails, DM multipath switches from the active path to the passive path. DM multipath can be configured in an active-active mode, where I/O is spread over the paths in a round-robin fashion. In some configurations, DM multipath can detect loading on the I/O paths and dynamically rebalance the load.

The following steps explain how to enable and configure DM multipath for active-passive in a Red Hat host initiator:

1. Use the following command to check whether the system includes the DM multipath package:

```
[root@storage]# rpm -q device-mapper-multipath
device-mapper-multipath-0.8.7-20.el9.x86_64
```

   Check the output in the Initiator Host used in this setup that displays the version of the DM multipath.

2. If the system does not include the package, use the following command to install it:

```
dnf install device-mapper-multipath
```

3. Use the following command to enable and initialize the multipath configuration file:

```
[root@storage]# mpathconf --enable
[root@storage]#
[root@storage]# cat /etc/multipath.conf
# device-mapper-multipath configuration file
# For a complete list of the default configuration values, run either:
# # multipath -t
# or
# # multipathd show config
# For a list of configuration options with descriptions, see the
# multipath.conf man page.
defaults {
user_friendly_names yes
```

```
find_multipaths yes
}
blacklist {
```

This will initialize the configuration file at **etc/multipath.conf** that displays the default setup.

4. Use the **multipath -t** or **multipath show config** commands to see the complete list of default configuration:

```
[root@storage]# multipath -t
defaults {
verbosity 2
polling_interval 5
max_polling_interval 20
reassign_maps "no"
multipath_dir "/lib64/multipath"
path_selector "service-time 0"
path_grouping_policy "failover"
uid_attribute "ID_SERIAL"
prio "const"
prio_args ""
features "0"
…
…
***Output Omitted**
```

5. Use the following command to start the DM multipath daemon and confirm whether the DM multipath daemon is running without issues. Use the following command to verify whether the multipath driver is loaded, enabled, and successfully started.

```
[root@storage]# systemctl start multipathd.service
[root@storage]#
[root@storage]# systemctl status multipathd.service
● multipathd.service - Device-Mapper Multipath Device Controller
Loaded: loaded (/usr/lib/systemd/system/multipathd.service; enabled; preset:
enabled)
Active: active (running) since Mon 2024-05-13 17:47:07 EDT; 26s ago
TriggeredBy: ○ multipathd.socket
Process: 11757 ExecStartPre=/sbin/modprobe -a scsi_dh_alua scsi_dh_emc scsi_
dh_rdac dm-multipath
(code=exited, status=0/SUCCESS)
Process: 11758 ExecStartPre=/sbin/multipath -A (code=exited, status=0/SUCCESS)
Main PID: 11759 (multipathd)
Status: "up"
Tasks: 7
Memory: 19.2M
CPU: 29ms
CGroup: /system.slice/multipathd.service
└─11759 /sbin/multipathd -d -s
May 13 17:47:07 storage.com systemd[1]: Starting Device-Mapper Multipath
Device Controller...
May 13 17:47:07 storage.com multipathd[11759]: --------start up--------
May 13 17:47:07 storage.com multipathd[11759]: read /etc/multipath.conf
May 13 17:47:07 storage.com multipathd[11759]: path checkers start up
```

```
May 13 17:47:07 storage.com systemd[1]: Started Device-Mapper Multipath Device
Controller.
```

6. Use the following command to verify the name of device mapper multipath:

```
[root@storage]# ls /dev/mapper
control   mpatha   rhel-home   rhel-root   rhel-swap
```

A new multipath has an alias of “mpatha", By default, any new multipath alias is created by adding to “mpath” the sequence of the alphabet starting with “a”. You can modify the name of the multipath by editing the config file.

7. At this point, discovery and login to iSCSI Target is completed. For more information, see Configuring iSCSI Initiator on NVIDIA (Mellanox) Ethernet Smart NIC on Red Hat. If a folder is mounted, use the **unmount** command to unmount it. This is necessary because the multipath device mapper will use two virtual disk as targets instead of one.
8. Use the following command to confirm that two target paths are logged:

```
[root@storage]# iscsiadm -m node
192.168.1.10:3260,1 iqn.2003-01.org.linux-iscsi.dl360_
target.x8664:sn.04eaf6dd4e42
192.168.2.10:3260,1 iqn.2003-01.org.linux-iscsi.dl360_
target.x8664:sn.04eaf6dd4e42
```

9. Login again to targets or reconfigure nodes. Use the **systemctl restart multipathd.service** command to restart the multipathd.service to recognize new paths.
10. To list the available disks in the initiator, use the **lsblk** command:

Two virtual disks are mapped to the same multipath device. At target side, that could be any target (storage array) or a Linux server, a single disk or partition is represented as 2 devices in the initiator.

```
[root@storage]# systemctl restart multipathd.service
[root@storage]#
[root@storage]# lsblk
NAME          MAJ:MIN RM  SIZE RO TYPE  MOUNTPOINTS
sda             8:0    0  2.2T  0 disk
├─sda1          8:1    0  600M  0 part  /boot/efi
├─sda2          8:2    0    1G  0 part  /boot
└─sda3          8:3    0  2.2T  0 part
├─rhel-root 253:0    0   70G  0 lvm   /
├─rhel-swap 253:1    0 31.3G  0 lvm   [SWAP]
└─rhel-home 253:2    0  2.1T  0 lvm   /home
sdb             8:16   1    0B  0 disk
sdc             8:32   0 36.5G  0 disk
└─mpatha      253:3    0 36.5G  0 mpath
sdd             8:48   0 36.5G  0 disk
```

```
└─mpatha      253:3    0 36.5G  0 mpath
```

11. At this point, there is no local folder mounted to the target disk. As shown below with the **df -h** command, there is still no association of the alias of the multipath device to the corresponding device mapper. Format the device mapper multipath and create a folder with **mkfs** command to mount it.

```
[root@storage]# df -h
Filesystem             Size  Used Avail Use% Mounted on
devtmpfs               4.0M     0  4.0M   0% /dev
tmpfs                   32G     0   32G   0% /dev/shm
tmpfs                   13G   11M   13G   1% /run
/dev/mapper/rhel-root   70G   34G   37G  48% /
/dev/sda2             1014M  393M  622M  39% /boot
/dev/sda1              599M  7.0M  592M   2% /boot/efi
/dev/mapper/rhel-home  2.1T   16G  2.1T   1% /home
tmpfs                  6.3G   52K  6.3G   1% /run/user/42
tmpfs                  6.3G   36K  6.3G   1% /run/user/1000
```

```
[root@storage]# mkfs.xfs -f /dev/mapper/mpatha
meta-data=/dev/mapper/mpatha     isize=512    agcount=16, agsize=597440 blks
=                       sectsz=4096  attr=2, projid32bit=1
=                       crc=1        finobt=1, sparse=1, rmapbt=0
=                       reflink=1    bigtime=1 inobtcount=1
data =                       bsize=4096   blocks=9559040, imaxpct=25
=                       sunit=64     swidth=64 blks
naming   =version 2              bsize=4096   ascii-ci=0, ftype=1
log      =internal log           bsize=4096   blocks=4667, version=2
```

```
[root@storage]# mkdir /Target_MP
[root@storage]#
[root@storage]# mount /dev/mapper/mpatha /Target_MP
[root@storage]#
[root@storage]# df -h
Filesystem             Size  Used Avail Use% Mounted on
devtmpfs               4.0M     0  4.0M   0% /dev
tmpfs                   32G     0   32G   0% /dev/shm
tmpfs                   13G   11M   13G   1% /run
/dev/mapper/rhel-root   70G   34G   37G  48% /
/dev/sda2             1014M  393M  622M  39% /boot
/dev/sda1              599M  7.0M  592M   2% /boot/efi
/dev/mapper/rhel-home  2.1T   16G  2.1T   1% /home
tmpfs                  6.3G   52K  6.3G   1% /run/user/42
tmpfs                  6.3G   36K  6.3G   1% /run/user/1000
/dev/mapper/mpatha      37G  294M   37G   1% /Target_MP
```

12. Use the **fdisk** command to check the output of and review the details of the remote volume attached to the initiator:

```
[root@storage]# fdisk -l | sed -n '/mpatha/,$p'
```

```
Disk /dev/mapper/mpatha: 36.46 GiB, 39153827840 bytes, 76472320 sectors
Units: sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 4096 bytes
I/O size (minimum/optimal): 262144 bytes / 262144 bytes
```

13. Use the **multipath -ll** command to review the multipath topology:

```
[root@storage]# multipath -ll
mpatha (360014055af5e5e46f0b4e76bf0f24d39) dm-3 LIO-ORG,disk1
size=36G features='0' hwhandler='1 alua' wp=rw
|-+- policy='service-time 0' prio=50 status=active
| `- 2:0:0:0 sdc 8:32 active ready running
`-+- policy='service-time 0' prio=50 status=enabled
`- 3:0:0:0 sdd 8:48 active ready running
```

In the example above, of the two available paths (active, ready, running), the one associated with the local disk SDC is in an active state. The other path is ready with a status of enabled, indicating it is the passive path. When the active path fails, the passive path becomes active.

14. To change to the active-active mode, modify **multipath.conf** file by adding the parameter **path_grouping_policy** to be **multibus**:

```
root@storage]# cat /etc/multipath.conf
# device-mapper-multipath configuration file
# For a complete list of the default configuration values, run either:
# # multipath -t
# or
# # multipathd show config
# For a list of configuration options with descriptions, see the
# multipath.conf man page.
defaults {
user_friendly_names yes
find_multipaths yes
path_grouping_policy multibus
}
blacklist {
#       devnode ".*"
```

The default value for this parameter is failover, which enables the active-standby mode.

15. Restart the multipath deamon and check again the multipath topology. The two available paths are now within the same group, sharing the same active status. With default seetings, any I/O operation will split the transfer of data between the two available paths due to the current policy applied, service-time 0. To modify the policy applied to distribute the data transfer across the available paths with a different method, use the **path_selector** parameter with available options.

```
[root@storage]# systemctl restart multipathd.service
[root@storage]#
```

```
[root@storage]# multipath -ll
mpatha (360014055af5e5e46f0b4e76bf0f24d39) dm-3 LIO-ORG,disk1
size=36G features='0' hwhandler='1 alua' wp=rw
`-+- policy='service-time 0' prio=50 status=active
|- 2:0:0:0 sdc 8:32 active ready running
`- 3:0:0:0 sdd 8:48 active ready running
```

16. Use the following commands to enable LLFC of NVIDIA (Mellanox) NICs in the Red Hat host and enable reception and transmission of pause frames if those are off.

```
[root@storage]# ethtool -A ens3f0np0 tx on rx on
[root@storage]# ethtool -a ens3f0np0
Pause parameters for ens3f0np0:
Autonegotiate:  off
RX:             on
TX:             on
```

17. Use the following command to confirm that supported pause frame use is set to Symmetric.

```
[root@storage]# ethtool  ens3f0np0 | grep pause
Supported pause frame use: Symmetric
Advertised pause frame use: Symmetric
Link partner advertised pause frame use: No
```

The above example demonstrates one of the 100 G interfaces used in the Initiator.

18. For scenarios with oversubscription or traffic congestion, check the flow control on the switch to confirm if pause frames are sent to an Initiator as shown in the Configuring LLFC. The pause frames will be sent to Initiator if the switch detects the oversubscription in the ports that provides connectivity to the Target. Confirm that the pause frames sent by switch are received, with **ethtool --include-statistics -a interface** command.

    The following is an output example that shows pause frames received with a condition of overload on the Target ports.

```
root@storage]# ethtool --include-statistics -a ens3f0np0
Pause parameters for ens3f0np0:
Autonegotiate:  off
RX:             on
TX:             on
Statistics:
tx_pause_frames: 0
rx_pause_frames: 2140
```

Multiple options are available to tune, modify, or update multipath features from the initiator's hosts. For more information, see Red Hat - Configure Device Mapper Multipath.

Configuring PFC on Nvidia (Mellanox) NIC Adapters

To configure PFC on Nvidia (Mellanox) NIC adapters, complete the following steps:

1. Use the **mlnx_qos -i interface** command to review current settings per interfaces.

```
[root@storage]# mlnx_qos -i ens3f0np0
DCBX mode: OS controlled
Priority trust state: pcp
default priority:
Receive buffer size (bytes): 20016,156096,0,0,0,0,0,0,
Cable len: 7
PFC configuration:
priority    0   1   2   3   4   5   6   7
enabled     0   0   0   0   0   0   0   0
buffer      0   0   0   0   0   0   0   0
tc: 1 ratelimit: unlimited, tsa: vendor
priority:  0
tc: 0 ratelimit: unlimited, tsa: vendor
priority:  1
tc: 2 ratelimit: unlimited, tsa: vendor
priority:  2
tc: 3 ratelimit: unlimited, tsa: vendor
priority:  3
tc: 4 ratelimit: unlimited, tsa: vendor
priority:  4
tc: 5 ratelimit: unlimited, tsa: vendor
priority:  5
tc: 6 ratelimit: unlimited, tsa: vendor
priority:  6
tc: 7 ratelimit: unlimited, tsa: vendor
priority:  7
```

- After MLNX OFED is installed, you can review, check, and modify PFC setup at Linux Initiators.
- **mlnx_qos -I** interface provides information about the current trust state (PCP or DSCP) and PFC configuration matrix, besides DCBx mode.

2. Use the **mlnx_qos -i interf --pfc** command to enable the desired priority.

All the eight priorities are disabled by default.

This scenario is configuring and using Priority 4 for iSCSI traffic. To enable the corresponding priority, change the value to 1 at position 4, with the count starting from 0. Use **--pfc 0,0,0,0,1,0,0,0** to enable priority 4 as shown in the following example.

Enabled and buffer values are shown now with value 1 as corresponding. The outputs are just showing only setup for port ens3f0np0, this should be also configured at interface ens3f1np1.

```
[root@storage]#  mlnx_qos -i ens3f0np0 --pfc 0,0,0,0,1,0,0,0
DCBX mode: OS controlled
Priority trust state: pcp
default priority:
Receive buffer size (bytes): 20016,156096,0,0,0,0,0,0,
```

```
Cable len: 7
PFC configuration:
 priority    0   1   2   3   4   5   6   7
 enabled     0   0   0   0   1   0   0   0
 buffer      0   0   0   0   1   0   0   0
tc: 1 ratelimit: unlimited, tsa: vendor
priority:  0
tc: 0 ratelimit: unlimited, tsa: vendor
priority:  1
tc: 2 ratelimit: unlimited, tsa: vendor
priority:  2
tc: 3 ratelimit: unlimited, tsa: vendor
priority:  3
tc: 4 ratelimit: unlimited, tsa: vendor
priority:  4
tc: 5 ratelimit: unlimited, tsa: vendor
priority:  5
tc: 6 ratelimit: unlimited, tsa: vendor
priority:  6
tc: 7 ratelimit: unlimited, tsa: vendor
priority:  7
```

3. Use the **mlnx_qos -i interf -trust pcp/dscp** command to change the value of the priority trust state. Modifying the priority trust state is important to synchronize how the system will treat marking values across the network.

```
[root@storage]# mlnx_qos -i ens3f0np0 --trust dscp
DCBX mode: OS controlled
Priority trust state: dscp
dscp2prio mapping:
prio:0 dscp:07,06,05,04,03,02,01,00,
prio:1 dscp:15,14,13,12,11,10,09,08,
prio:2 dscp:23,22,21,20,19,18,17,16,
prio:3 dscp:31,30,29,28,27,26,25,24,
prio:4 dscp:39,38,37,36,35,34,33,32,
prio:5 dscp:47,46,45,44,43,42,41,40,
prio:6 dscp:55,54,53,52,51,50,49,48,
prio:7 dscp:63,62,61,60,59,58,57,56,
…
…
Output Omitted
```

4. Use the **cat /proc/net/vlan/vlan<ID>** command to check the status of Ingress or Egress priority mappings:

```
[root@storage]# cat /proc/net/vlan/vlan10
vlan10  VID: 10  REORDER_HDR: 1  dev->priv_flags: 81021
total frames received          277
total bytes received         25262
Broadcast/Multicast Rcvd         0
total frames transmitted       437
total bytes transmitted      43841
Device: ens3f0np0
```

```
INGRESS priority mappings: 0:0  1:0  2:0  3:0  4:0  5:0  6:0 7:0
EGRESS priority mappings:
```

Egress priority mapping is part of a proper design to mark, classify, and filter packets according to a design that permits the management of a congested network and avoids losses. The PFC priority map correlates with the egress priority mapping matrix, which manages eight queues to classify or assign different kinds of traffic.

5. To change the values of Egress priority, use the **ip link set** command:

```
[root@storage]# ip link set vlan10 type vlan egress 0:4
Device: ens3f0np0
INGRESS priority mappings: 0:0  1:0  2:0  3:0  4:0  5:0  6:0 7:0
[root@storage] /]# cat /proc/net/vlan/vlan10
vlan10  VID: 10  REORDER_HDR: 1  dev->priv_flags: 81021
total frames received          931
total bytes received        84368
Broadcast/Multicast Rcvd            0
total frames transmitted         1390
total bytes transmitted       133103
Device: ens3f0np0
INGRESS priority mappings: 0:0  1:0  2:0  3:0  4:0  5:0  6:0 7:0
EGRESS priority mappings: 0:4
```

Consider that IP links command changes might be non-persistent, validate the changes with your current RHEL version.

6. Verify flow control in the switch to confirm if pause frames are sent to an Initiator. For more information, see Configuring PFC .

Pause frames will be sent to the Initiator if the switch detects the over subscription in the port that provides connectivity to the target.

7. Use the **ethtool -S <interface>** command to verify flow control in the Initiator to confirm that pause frames sent by the switch are received:

```
[root@storage]# ethtool -S ens3f0np0 | grep pause
rx_pause_ctrl_phy: 15592
tx_pause_ctrl_phy: 0
rx_prio4_pause: 15592
rx_prio4_pause_duration: 456194
tx_prio4_pause: 0
tx_prio4_pause_duration: 0
rx_prio4_pause_transition: 77085
tx_pause_storm_warning_events: 0
tx_pause_storm_error_events: 0
```

Configuring PFC with DCBx on Nvidia (Mellanox) NIC Adapters

To configure PFC with DCBx on Nvidia (Mellanox) NIC adapters, complete the following steps:

1. Use the following command to start Nvidia (Mellanox) software tools to enable the application that can be used to verify, review, and modify NIC parameters like PFC and DCBx. The following shows the expected output after starting mst modules:

```
[root@storage]# mst start
Starting MST (Mellanox Software Tools) driver set
Loading MST PCI module - Success
Loading MST PCI configuration module - Success
Create devices
Unloading MST PCI module (unused) - Success
```

Enabling DCBx in the host initiatior permits to avoid manual setup of PFC. PFC parameters will be learn from the HPE ANW CX 9300 Switch Series. This is separate from the **mlnx_qos** tool, but both will be available after installing OFED Tools from Nvidia (Mellanox) NIC adapters.

2. Use the following command to verify the status of mst:

```
[root@storage]# mst status
MST modules:
------------
MST PCI module is not loaded
MST PCI configuration module loaded
MST devices:
------------
/dev/mst/mt4117_pciconf0         - PCI configuration cycles access.
domain:bus:dev.fn=0000:5d:00.0 addr.reg=88 data.reg=92 cr_bar.gw_offset=-1
Chip revision is: 00
/dev/mst/mt4119_pciconf0         - PCI configuration cycles access.
domain:bus:dev.fn=0000:12:00.0 addr.reg=88 data.reg=92 cr_bar.gw_offset=-1
Chip revision is: 00
/dev/mst/mt4125_pciconf0         - PCI configuration cycles access.
domain:bus:dev.fn=0000:d8:00.0 addr.reg=88 data.reg=92 cr_bar.gw_offset=-1
Chip revision is: 00
```

The application shows devices detected according to cards installed in the host. If server uses a dual port card, it is expected that it will only detect one PCI device.

3. To confirm which card the system uses, use the following command to list the connected PCI devices:

```
lspci | grep Mellanox
```

Check the output of the **lspci** command using a filter for Nvidia (Mellanox) PCI cards to confirm bus IDs. For this scenario, the ConnectX-6Dx Mellanox card is used.

```
[root@storage]# lspci | grep Mellanox
12:00.0 Ethernet controller: Mellanox Technologies MT27800 Family [ConnectX-5]
12:00.1 Ethernet controller: Mellanox Technologies MT27800 Family [ConnectX-5]
5d:00.0 Ethernet controller: Mellanox Technologies MT27710 Family [ConnectX-4
Lx]
5d:00.1 Ethernet controller: Mellanox Technologies MT27710 Family [ConnectX-4
Lx]
d8:00.0 Ethernet controller: Mellanox Technologies MT2892 Family [ConnectX-6
Dx]
d8:00.1 Ethernet controller: Mellanox Technologies MT2892 Family [ConnectX-6
Dx]
```

4. Each Ethernet Smart NIC can be modified with **mlxconfig** tool. Use a filter to list LLDP and DCBX capabilities of Nvidia (Mellanox) card as shown below.

```
[root@storage]# mlxconfig -d /dev/mst/mt4125_pciconf0 q | grep LLDP
LLDP_NB_DCBX_P1                             False(0)
LLDP_NB_RX_MODE_P1                          OFF(0)
LLDP_NB_TX_MODE_P1                          OFF(0)
LLDP_NB_DCBX_P2                             False(0)
LLDP_NB_RX_MODE_P2                          OFF(0)
LLDP_NB_TX_MODE_P2                          OFF(0)
[root@storage]# mlxconfig -d /dev/mst/mt4125_pciconf0 q | grep DCBX
LLDP_NB_DCBX_P1                             False(0)
LLDP_NB_DCBX_P2                             False(0)
DCBX_IEEE_P1                                True(1)
DCBX_CEE_P1                                 True(1)
DCBX_WILLING_P1                             True(1)
DCBX_IEEE_P2                                True(1)
DCBX_CEE_P2                                 True(1)
DCBX_WILLING_P2                             True(1)
```

Depending on the NIC version, software upgrade, or device family, the default may vary. In this case, the default value is false, so it will be modified. DCBx has three configurable items in this PCI card. It is possible to enable DCBx IEEE mode, DCBX cee mode, or both, and enable the willing bit to accept PFC configuration sent from the switch. If the default DCBx version on the switch is activated (DCBx IEEE mode), confirm that at least IEEE mode is enabled to permit proper synchronization.

5. To change default values of LLDP or DCBx parameters, use the following command with **set** option of the **mlxconfig** tool:

```
[root@storage]# mlxconfig -d /dev/mst/mt4125_pciconf0 set LLDP_NB_DCBX_P1=TRUE
LLDP_NB_DCBX_P2=TRUE
Device #1:
----------
Device type:    ConnectX6DX
Name:           MCX623106AS-CDA_Ax
Description:    ConnectX-6 Dx EN adapter card; 100GbE; Dual-port QSFP56; PCIe
4.0 x16; Secure Boot; No Crypto
Device:         /dev/mst/mt4125_pciconf0
Configurations:                                     Next Boot       New
LLDP_NB_DCBX_P1                             False(0)        True(1)
```

```
LLDP_NB_DCBX_P2                             False(0)        True(1)
Apply new Configuration? (y/n) [n] : y
Applying... Done!
-I- Please reboot machine to load new configurations.
[root@storage]# mlxconfig -d /dev/mst/mt4125_pciconf0 set LLDP_NB_RX_MODE_P1=2
LLDP_NB_TX_MODE_P1=2
Device #1:
----------
Device type:    ConnectX6DX
Name:           MCX623106AS-CDA_Ax
Description:    ConnectX-6 Dx EN adapter card; 100GbE; Dual-port QSFP56; PCIe
4.0 x16; Secure Boot; No Crypto
Device:         /dev/mst/mt4125_pciconf0
Configurations:                                         Next Boot       New
LLDP_NB_RX_MODE_P1                          OFF(0)          ALL(2)
LLDP_NB_TX_MODE_P1                          OFF(0)          ALL(2)
Apply new Configuration? (y/n) [n] : Y
Applying... Done!
-I- Please reboot machine to load new configurations.
[root@storage]# mlxconfig -d /dev/mst/mt4125_pciconf0 set LLDP_NB_RX_MODE_P2=2
LLDP_NB_TX_MODE_P2=2
Device #1:
----------
Device type:    ConnectX6DX
Name:           MCX623106AS-CDA_Ax
Description:    ConnectX-6 Dx EN adapter card; 100GbE; Dual-port QSFP56; PCIe
4.0 x16; Secure Boot; No Crypto
Device:         /dev/mst/mt4125_pciconf0
Configurations:                                         Next Boot       New
LLDP_NB_RX_MODE_P2                          OFF(0)          ALL(2)
LLDP_NB_TX_MODE_P2                          OFF(0)          ALL(2)
Apply new Configuration? (y/n) [n] : y
Applying... Done!
-I- Please reboot machine to load new configurations.
```

Each parameter is assigned a new value. You can test immediately the changes but is recommended to reboot the machine to make the new values permanent.

6. Use the following command to check again to confirm whether the values are changed as requested:

```
[root@storage]# mlxconfig -d /dev/mst/mt4119_pciconf0 q | grep LLDP
LLDP_NB_DCBX_P1                             True(1)
LLDP_NB_RX_MODE_P1                          ALL(2)
LLDP_NB_TX_MODE_P1                          ALL(2)
LLDP_NB_DCBX_P2                             True(1)
LLDP_NB_RX_MODE_P2                          ALL(2)
LLDP_NB_TX_MODE_P2                          ALL(2)
```

7. Use the following command to reset the Nvidia (Mellanox) firmware:

```
[root@storage]# mlxfwreset -d /dev/mst/mt4119_pciconf0 --level 3 reset


Requested reset level for device, /dev/mst/mt4125_pciconf0:

3: Driver restart and PCI reset
Continue with reset?[y/N] y
-I- Sending Reset Command To Fw             -Done
-I- Stopping Driver                         -Done
-I- Resetting PCI                           -Done
-I- Starting Driver                         -Done
-I- Restarting MST                          -Done
-I- FW was loaded successfully.
```

8. Optionally, use the following command to change the DCBx Mode to Firmware Controlled so the NIC will have governance of the DBCx controls, which means changes to QoS settings cannot be done by the host operating system.

```
root@storage]#  mlnx_qos -i ens3f0np0 -d fw
DCBX mode: Firmware controlled
Priority trust state: pcp
default priority:
**
**
Output Omitted
```

9. Use the following command to check the output of the **mlnx_qos -i** command again for any interface of the multipath and confirm the new PFC Configuration:

```
[root@storage]# mlnx_qos -i ens3f0np0 -d fw
****** WARNING: lldpad service is running and may overwrite your settings
******

DCBX mode: Firmware controlled
Priority trust state: pcp
default priority:
Receive buffer size (bytes): 20016,156096,0,0,0,0,0,0,
Cable len: 7
PFC configuration:
priority    0   1   2   3   4   5   6   7
enabled   0   0   0   0   1   0   0   0
buffer      0   0   0   0   1   0   0   0
tc: 0 ratelimit: unlimited, tsa: ets, bw: 20%
priority:  0
priority:  1
priority:  2
priority:  3
priority:  5
tc: 1 ratelimit: unlimited, tsa: ets, bw: 80%
priority:  4
tc: 2 ratelimit: unlimited, tsa: strict
priority:  6
```

As application iSCSI with priority 4 is setup at the switch, the enabled and buffer bit are now shown in 1.

10. Use the following command to check and review the status of the PFC setup. Confirm priority mapping, TC Bandwidth, and PFC TLV parameters.

```
[root@storage]# lldptool -t -i ens3f0np0
Chassis ID TLV
MAC: 88:e9:a4:8e:60:46
Port ID TLV
MAC: 88:e9:a4:8e:60:47
Time to Live TLV
120
IEEE 8021QAZ ETS Configuration TLV
Willing: yes
CBS: not supported
MAX_TCS: 8
PRIO_MAP: 0:0 1:0 2:0 3:0 4:1 5:0 6:2 7:2
TC Bandwidth: 20% 80% 0% 0% 0% 0% 0% 0%
TSA_MAP: 0:ets 1:ets 2:strict 3:ets 4:ets 5:ets 6:ets 7:ets
IEEE 8021QAZ PFC TLV
Willing: yes
MACsec Bypass Capable: no
PFC capable traffic classes: 8
PFC enabled: 4
End of LLDPDU TLV
```

11. Use the following command to confirm PFC parameters received from switches. Verify whether the host receives the application TLV for iSCSI, by checking the corresponding TCP port (860 and 3260).

This is informational to the host, and additional setup will be necessary to tag frames with a PCP value of 4.

```
[root@storage]# lldptool get-tlv -n -i ens3f0np0
Chassis ID TLV
MAC: 18:7a:3b:74:6f:00
Port ID TLV
Ifname: 1/1/25
Time to Live TLV
120
System Name TLV
9300-CoreA
System Description TLV
Aruba R8Z96A  CL.10.13.1000
System Capabilities TLV
System capabilities:  Bridge, Router
Enabled capabilities: Bridge, Router
Management Address TLV
IPv4: 192.168.1.15
Ifindex: 16777226
OID: 0.43.6.1.2.1.31.1.1.1.1.-120.-128.-128.10
Port Description TLV
description Host_Initiator_iSCSI_Vlan10
IEEE 8021QAZ PFC TLV
```

```
Willing: no
MACsec Bypass Capable: no
PFC capable traffic classes: 7
PFC enabled: 4
IEEE 8021QAZ ETS Configuration TLV
Willing: no
CBS: not supported
MAX_TCS: 3
PRIO_MAP: 0:0 1:0 2:0 3:0 4:1 5:0 6:2 7:2
TC Bandwidth: 20% 80% 0% 0% 0% 0% 0% 0%
TSA_MAP: 0:ets 1:ets 2:strict 3:ets 4:ets 5:ets 6:ets 7:ets
IEEE 8021QAZ ETS Recommendation TLV
PRIO_MAP:  0:0 1:0 2:0 3:0 4:1 5:0 6:2 7:2
TC Bandwidth: 20% 80% 0% 0% 0% 0% 0% 0%
TSA_MAP: 0:ets 1:ets 2:strict 3:ets 4:ets 5:ets 6:ets 7:ets
IEEE 8021QAZ APP TLV
App#   0         Prio 4          Sel 2   P-ID    {S}TCP Port: 860
App#   1         Prio 4          Sel 2   P-ID    {S}TCP Port: 3260
Port VLAN ID TLV
PVID: 10
VLAN Name TLV
VID 10: Name VLAN10
Link Aggregation TLV
Aggregation capable
Currently aggregated
Aggregated Port ID: 1
Maximum Frame Size TLV
9198
Unidentified Org Specific TLV
OUI: 0x883a30, Subtype: 2, Info: 0001
End of LLDPDU TLV
```

Multipath Diagnostic and Failover Validation

To diagnose and validate failover, complete the following steps:

1. Validate the new iSCSI disk using a test file by opening the local folder associated to the Target with **cd Target_MP**. In the following example, the **dd** program is used to transfer a file to the folder, which uses iSCSI to transfer files to the Target. This is a transfer of a test file of 1 GB, registering the time to complete the transfer and the average bandwidth to complete the operation. Confirm that the file has been saved in the local folder.

```
[root@storage]# cd Target_MP
[root@storage]# Target_MP]#
[root@storage Target_MP]# dd if=/dev/random of=test1G1 bs=1M count=1000
1000+0 records in
1000+0 records out
1048576000 bytes (1.0 GB, 1000 MiB) copied, 2.52641 s, 415 MB/s
[root@storage Target_MP]#
[root@storage Target_MP]# ls -lh
total 21G
drwxr-xr-x. 2 root root     6 May 14 12:17 ftest1
-rw-r--r--. 1 root root     0 May 14 12:17 test1
-rw-r--r--. 1 root root 1000M May 28 18:11 test1G1
-rw-r--r--. 1 root root 1000M May 14 12:27 test1G2
```

2. To confirm that DM multipath is working properly, perform a failover test to check resilience of the paths to the Target. To perform this task, use the **multipath -ll** command to check current multipath topology to track the active path. The **ip a** command with a grep filter is used as shown below to get the ip addressing of the multipath links.

3. To simulate a failure in the network, shutdown the switch port that connects to Initiator active NIC interface.

   For this example, port 1/1/25 at Core A is shutdown to validate multipath functionality. The shutdown interface reports failed faulty running and status is active. The multipath status of the path shutdown keeps the active status until new data is sent to the Target. When data is sent, the interface that is shutdown has a status of enabled, and the passive interface status goes to active, showing DM multipath is providing active-passive resiliency.

```
[root@storage Target_MP]# ip a | grep 'ens3\|inet 192'
6: ens3f0np0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc mq state UP
group
inet 192.168.1.11/24 brd 192.168.1.255 scope global noprefixroute ens3f0np0
7: ens3f1np1: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc mq state UP
group
inet 192.168.2.11/24 brd 192.168.2.255 scope global noprefixroute ens3f1np1
[root@storage Target_MP]#
[root@storage Target_MP]# multipath -ll
mpatha (360014055af5e5e46f0b4e76bf0f24d39) dm-3 LIO-ORG,disk1
size=36G features='0' hwhandler='1 alua' wp=rw
|-+- policy='service-time 0' prio=50 status=active
| `- 2:0:0:0 sdc 8:32 active ready running
`-+- policy='service-time 0' prio=50 status=enabled
`- 3:0:0:0 sdd 8:48 active ready running
[root@storage Target_MP]#
-------------------------------------------------------------------------------
------------------
Core-A#config
Core-A(config)#int 1/1/25
Core-A(config-if)#shut
-------------------------------------------------------------------------------
------------------
[root@storage Target_MP]# ip a | grep ens3f0np0
6: ens3f0np0: <BROADCAST,MULTICAST> mtu 1500 qdisc mq state DOWN group default
qlen 1000
[root@storage Target_MP]# multipath -ll
mpatha (360014055af5e5e46f0b4e76bf0f24d39) dm-3 LIO-ORG,disk1
size=36G features='0' hwhandler='1 alua' wp=rw
|-+- policy='service-time 0' prio=0 status=active
| `- 2:0:0:0 sdc 8:32 failed faulty running
`-+- policy='service-time 0' prio=50 status=enabled
`- 3:0:0:0 sdd 8:48 active ready running
[root@storage Target_MP]#
[root@storage Target_MP]# dd if=/dev/random of=test1G6 bs=1M count=1000
1000+0 records in
1000+0 records out
1048576000 bytes (1.0 GB, 1000 MiB) copied, 2.5258 s, 415 MB/s
[root@storage Target_MP]#
[root@storage Target_MP]# multipath -ll
mpatha (360014055af5e5e46f0b4e76bf0f24d39) dm-3 LIO-ORG,disk1
size=36G features='0' hwhandler='1 alua' wp=rw
|-+- policy='service-time 0' prio=0 status=enabled
| `- 2:0:0:0 sdc 8:32 failed faulty running
```

```
`-+- policy='service-time 0' prio=50 status=active
`- 3:0:0:0 sdd 8:48 active ready running
```

4. To test active-active mode, use the **multipath -ll** command to test with the **dd** program to transfer a file, and confirm that file is saved in the local folder. Check the queues in the switches ports that connect to the Target to confirm that data transfer was split using both paths to save the data. The following example shows that a transfer of a 3G file, uses both paths splitting almost by the half of the size between them.

```
[root@storage Target_MP]# multipath -ll
mpatha (360014055af5e5e46f0b4e76bf0f24d39) dm-3 LIO-ORG,disk1
size=36G features='0' hwhandler='1 alua' wp=rw
`-+- policy='service-time 0' prio=50 status=active
|- 2:0:0:0 sdc 8:32 active ready running
`- 3:0:0:0 sdd 8:48 active ready running
-------------------------------------------------------------------------------
---
[root@storageTarget_MP]# dd if=/dev/random of=test3G1 bs=1M count=3000
3000+0 records in
3000+0 records out
3145728000 bytes (3.1 GB, 2.9 GiB) copied, 7.56061 s, 416 MB/s
[root@storage Target_MP]#
-------------------------------------------------------------------------------
--

Core-A# show int 1/1/26 queues
Interface 1/1/26 is up
Admin state is up
Unicast
        Tx Bytes        Tx Packets          Tx Drops        Tx Byte Depth
Q0         0                0                0                254
Q1      1635308820          1025234          0               1245108
Q2         0                0                0                254
Q3         0                0                0                  0
Q4         0                0                0                  0
Q5         0                0                0                  0
Q6         0                0                0                  0
Q7         0                0                0                508
Multi-destination
         Tx Bytes        Tx Packets          Tx Drops        Tx Byte Depth
Q0         0                0                0               4165600
Q1         0                0                0               4167124
Q2         0                0                0                508
Q3         0                0                0                  0
-------------------------------------------------------------------------------
------

Core-B# show int 1/1/26 queues
Interface 1/1/26 is up
Admin state is up
Unicast
       Tx Bytes        Tx Packets          Tx Drops        Tx Byte Depth
Q0         0                0                0                2032
Q1     1664667424         1173012          0                2032
Q2         0                0                0                  0
```

```
Q3          0                  0                  0                  0
Q4          0                  0                  0                  0
Q5          0                  0                  0                  0
Q6          0                  0                  0                  0
Q7          0                  0                  0                508
Multi-destination
        Tx Bytes        Tx Packets          Tx Drops        Tx Byte Depth
Q0          0                  0                  0                254
Q1          0                  0                  0                  0
Q2          0                  0                  0                  0
Q3          0                  0                  0                  0
```

5. Repeat the failover test as done in step 2 to simulate a failure in the network, shut down the Core A switch port that connects to Initiator, port 1/1/25. Verify that new multipath status shows failed faulty running for the interface connected to the Core A and proceed to send a test file again to confirm that transfer functionality remains. The data transfer will occur using the active-ready path, as shown in the secondary switch, after completing the transfer of a 2G test file.

```
[root@storage Target_MP]# multipath -ll
mpatha (360014055af5e5e46f0b4e76bf0f24d39) dm-3 LIO-ORG,disk1
size=36G features='0' hwhandler='1 alua' wp=rw
`-+- policy='service-time 0' prio=50 status=active
|- 2:0:0:0 sdc 8:32 active ready running
`- 3:0:0:0 sdd 8:48 active ready running
------------------------------------------------------------------------------
Core-A(config)# int 1/1/25
Core-A(config-if)# shut
------------------------------------------------------------------------------
[root@storage Target_MP]# multipath -ll
mpatha (360014055af5e5e46f0b4e76bf0f24d39) dm-3 LIO-ORG,disk1
size=36G features='0' hwhandler='1 alua' wp=rw
`-+- policy='service-time 0' prio=50 status=active
|- 2:0:0:0 sdc 8:32 failed faulty running
`- 3:0:0:0 sdd 8:48 active ready running
[root@storage Target_MP]# dd if=/dev/random of=test2G2 bs=1M count=2000
2000+0 records in
2000+0 records out
2097152000 bytes (2.1 GB, 2.0 GiB) copied, 5.0353 s, 416 MB/s

------------------------------------------------------------------------------

Core-B# show int 1/1/26 queues
Interface 1/1/26 is up
Admin state is up
Unicast
       Tx Bytes        Tx Packets          Tx Drops          Tx Byte Depth
Q0        0                  0                  0                2032
Q1    2200314846       1460625               0                 2032

Q2        0                  0                  0                  0
Q3        0                  0                  0                  0
Q4        0                  0                  0                  0
Q5        0                  0                  0                  0
Q6        0                  0                  0                  0
Q7        0                  0                  0                508
Multi-destination
```

```
       Tx Bytes        Tx Packets          Tx Drops          Tx Byte Depth
Q0        0                0                  0                  0
Q1        0                0                  0                  0
Q2        0                0                  0                  0
Q3        0                0                  0                  0
```

Final iSCSI with LLFC and PFC Configuration on Air Gap Core Topology

This chapter includes the following topics:

LLFC Configuration Example on Air Gap Core

The following configuration example is the corresponding relevant configuration for a Air Gap Core that supports DM multipath and uses Link Level flow control as the mechanism to manage congestion.

Core-A with LLFC

```
!Core-A:
!***********
!Note: Only relevant configuration lines for this use case are shown
!
lldp dcbx
class ip tcp_iscsi_initiator
  10 match tcp any any eq iscsi count
class ip tcp_iscsi_target
  10 match tcp any eq iscsi any count
class ipv6 v6_tcp_iscsi_initiator
  10 match tcp any any eq iscsi count
class ipv6 v6_tcp_iscsi_target
  10 match tcp any eq iscsi any count
policy remark_iSCSI_Initiator
  10 class ip tcp_iscsi_initiator action pcp 4 action local-priority 4 action dscp
CS4
  20 class ipv6 v6_tcp_iscsi_initiator action pcp 4 action local-priority 4 action
dscp CS4
policy remark_iSCSI_Target
  10 class ip tcp_iscsi_target action pcp 4 action local-priority 4 action dscp CS4
  20 class ipv6 v6_tcp_iscsi_target action pcp 4 action local-priority 4 action dscp
CS4
!
vlan 10
  description iSCSI traffic vlan10
!
qos queue-profile lossless4p
  map queue 0 local-priority 0,1,2,3,5
  map queue 1 local-priority 4
    map queue 2 local-priority 6,7
qos schedule-profile lossless_sp4p
  dwrr queue 0 weight 20
  dwrr queue 1 weight 80
  strict queue 2
```

```
apply qos queue-profile lossless4p schedule-profile lossless_sp4p
qos trust dscp
qos pool 1 lossless size 50.00 percent headroom 5000 kbytes priorities 4
!
interface 1/1/25
  description Host_Initiator_iSCSI_Vlan10
  no shutdown
  mtu 9198
  flow-control rxtx pool 1 override-negotiation
  no routing
  vlan trunk native 1
  vlan trunk allowed 10
  apply policy remark_iSCSI_Initiator in
interface 1/1/26
  description Target_iSCSI_Vlan10
  no shutdown
  mtu 9198
  flow-control rxtx pool 1 override-negotiation
  no routing
  vlan trunk native 1
  vlan trunk allowed 10
  apply policy remark_iSCSI_Target in
!
interface vlan 10
  ip mtu 9198
  ip address 192.168.1.15/24
!
dcbx application iscsi priority 4
```

Core-B with LLFC

```
!Core-B:
!***********
!Note: Only relevant configuration lines for this use case are shown
!
lldp dcbx
class ip tcp_iscsi_initiator
  10 match tcp any any eq iscsi count
class ip tcp_iscsi_target
  10 match tcp any eq iscsi any count
class ipv6 v6_tcp_iscsi_initiator
  10 match tcp any any eq iscsi count
class ipv6 v6_tcp_iscsi_target
  10 match tcp any eq iscsi any count
policy remark_iSCSI_Initiator
  10 class ip tcp_iscsi_initiator action pcp 4 action local-priority 4 action dscp
CS4
  20 class ipv6 v6_tcp_iscsi_initiator action pcp 4 action local-priority 4 action
dscp CS4
policy remark_iSCSI_Target
  10 class ip tcp_iscsi_target action pcp 4 action local-priority 4 action dscp CS4
  20 class ipv6 v6_tcp_iscsi_target action pcp 4 action local-priority 4 action
dscp CS4
!
vlan 20
  description iSCSI traffic vlan20
qos queue-profile lossless4p
  map queue 0 local-priority 0,1,2,3,5
  map queue 1 local-priority 4
```

```
  map queue 2 local-priority 6,7
qos schedule-profile lossless_sp4p
  dwrr queue 0 weight 20
  dwrr queue 1 weight 80
  strict queue 2
apply qos queue-profile lossless4p schedule-profile lossless_sp4p
qos trust dscp
qos pool 1 lossless size 50.00 percent headroom 5000 kbytes priorities 4
!
interface 1/1/25
  description Host_Initiator_Vlan_20
  mtu 9198
  flow-control rxtx pool 1 override-negotiation
  no routing
  vlan trunk native 1
  vlan trunk allowed 20
  apply policy remark_iSCSI_Initiator in
interface 1/1/26
  description Target_iSCSI_Vlan_20
  no shutdown
  mtu 9198
  flow-control rxtx pool 1 override-negotiation
  no routing
  vlan trunk native 1
  vlan trunk allowed 20
  apply policy remark_iSCSI_Target in
!
interface vlan 20
  ip mtu 9198
  ip address 192.168.2.20/24
!
dcbx application iscsi priority 4
```

PFC Configuration Example on Air Gap Core

The following configuration example is the corresponding relevant configuration for a Air Gap Core that supports DM multipath and uses Priority-based Flow Control (PFC) as the mechanism to manage congestion.

Core-A with PFC

```
Core-A:
***********
!Note: Only relevant configuration lines for this use case are shown
!
lldp dcbx
!
class ip tcp_iscsi_initiator
  10 match tcp any any eq iscsi count
class ip tcp_iscsi_target
  10 match tcp any eq iscsi any count
class ipv6 v6_tcp_iscsi_initiator
  10 match tcp any any eq iscsi count
class ipv6 v6_tcp_iscsi_target
  10 match tcp any eq iscsi any count
policy remark_iSCSI_Initiator
```

```
  10 class ip tcp_iscsi_initiator action pcp 4 action local-priority 4 action dscp
CS4
  20 class ipv6 v6_tcp_iscsi_initiator action pcp 4 action local-priority 4 action
dscp CS4
policy remark_iSCSI_Target
  10 class ip tcp_iscsi_target action pcp 4 action local-priority 4 action dscp CS4
  20 class ipv6 v6_tcp_iscsi_target action pcp 4 action local-priority 4 action
dscp CS4
!
vlan 10
  description iSCSI traffic vlan10
!
qos queue-profile lossless4p
  map queue 0 local-priority 0,1,2,3,5
  map queue 1 local-priority 4
  map queue 2 local-priority 6,7
  qos schedule-profile lossless_sp4p
  dwrr queue 0 weight 20
  dwrr queue 1 weight 80
  strict queue 2
  apply qos queue-profile lossless4p schedule-profile lossless_sp4p
qos trust dscp
qos pool 1 lossless size 50.00 percent headroom 5000 kbytes priorities 4
!
interface 1/1/25
  description Host_Initiator_iSCSI_Vlan10
  no shutdown
  mtu 9198
  flow-control priority rxtx 4
  no routing
  vlan trunk native 1
  vlan trunk allowed 10
  apply policy remark_iSCSI_Initiator in
interface 1/1/26
  description Target_iSCSI_Vlan10
  no shutdown
  mtu 9198
  flow-control priority rxtx 4
  no routing
  vlan trunk native 1
  vlan trunk allowed 10
  apply policy remark_iSCSI_Target in
!
interface vlan 10
  ip mtu 9198
  ip address 192.168.1.15/24
!
dcbx application iscsi priority 4
```

Core-B with PFC

```
Core-B:
***********
!
!Note: Only relevant configuration lines for this use case are shown
!
lldp dcbx
!
class ip tcp_iscsi_initiator
```

```
  10 match tcp any any eq iscsi count
class ip tcp_iscsi_target
  10 match tcp any eq iscsi any count
class ipv6 v6_tcp_iscsi_initiator
  10 match tcp any any eq iscsi count
class ipv6 v6_tcp_iscsi_target
  10 match tcp any eq iscsi any count
policy remark_iSCSI_Initiator
  10 class ip tcp_iscsi_initiator action pcp 4 action local-priority 4 action dscp
CS4
  20 class ipv6 v6_tcp_iscsi_initiator action pcp 4 action local-priority 4 action
dscp CS4
policy remark_iSCSI_Target
  10 class ip tcp_iscsi_target action pcp 4 action local-priority 4 action dscp CS4
  20 class ipv6 v6_tcp_iscsi_target action pcp 4 action local-priority 4 action
dscp CS4
!
vlan 20
  description iSCSI traffic vlan20
qos queue-profile lossless4p
  map queue 0 local-priority 0,1,2,3,5
  map queue 1 local-priority 4
  map queue 2 local-priority 6,7
qos schedule-profile lossless_sp4p
  dwrr queue 0 weight 20
  dwrr queue 1 weight 80
  strict queue 2
apply qos queue-profile lossless4p schedule-profile lossless_sp4p
qos trust dscp
qos pool 1 lossless size 50.00 percent headroom 5000 kbytes priorities 4
!
interface 1/1/25
  description Host_Initiator_Vlan_20
  mtu 9198
  flow-control priority rxtx 4
  no routing
  vlan trunk native 1
  vlan trunk allowed 20
  apply policy remark_iSCSI_Initiator in
interface 1/1/26
  description Target_iSCSI_Vlan_20
  no shutdown
  mtu 9198
  flow-control priority rxtx 4
  no routing
  vlan trunk native 1
  vlan trunk allowed 20
  apply policy remark_iSCSI_Target in
!
interface vlan 20
  ip mtu 9198
  ip address 192.168.2.20/24
!
dcbx application iscsi priority 4
```