July 2018

Demartek

Demartek RoCE Deployment Guide 2018 Edition

Produced with the InfiniBand Trade Association in support of the RoCE Initiative.

INFINIBAND TRADE ASSOCIATION

RoCE

Executive Summary

Data center managers and administrators face increasing pressure to accelerate application performance because data continues to grow, and demand for quick access to this data continues to intensify.

The computing industry continues to develop new technologies such as flash storage and more powerful processors to keep pace with these demands. Product design engineers are also incorporating performance acceleration techniques into network adapters and switches.

One of these networking adapter technologies, Remote Direct Memory Access (RDMA), has been built into special network adapters that use RDMA over Converged Ethernet, or RoCE technology. This performance acceleration technology is built into specialized network adapters that work with a streamlined software stack in the operating system to improve performance and lower the host CPU consumption required to process network traffic.

The 2018 edition of the ***Demartek RoCE Deployment Guide*** is designed for managers and technical professionals within IT departments who are exploring the benefits of deploying RoCE technology or who are looking for practical guidance and deployment examples of RoCE solutions.

As RoCE includes server, adapter and switching technologies, this guide provides information and guidance in each area. A basic understanding of each of these areas is needed to successfully deploy RoCE technology.

RoCE is an industry standard driven by the InfiniBand Trade Association (IBTA).

This guide is intended to be used as a reference and includes screen shots and information from actual deployments of specific RoCE-enabled products.

All of the work was performed in the Demartek lab in Golden, Colorado, USA.

Demartek RoCE Deployment Guide 2018 Edition

Demartek
July 2018

RoCE Basic Definitions

RDMA – Remote Direct Memory Access - the remote memory management capability that allows server-to-server data movement directly between the application memory of each without any CPU involvement.

RoCE – (pronounced "Rocky") RDMA over Converged Ethernet – a network protocol that allows remote direct memory access over a converged Ethernet network.

Converged Ethernet – A term associated with a collection of standards-based extensions to traditional Ethernet that provide lossless characteristics that enable the convergence of Local Area Network (LAN) and Storage Area Network (SAN) technology onto a single unified fabric.

RoCE Initiative

The RoCE Initiative is an education and resource program of the InfiniBand Trade Association (IBTA) that is committed to increasing awareness of RoCE by providing technical education and reference solutions for high-performance Ethernet topologies in traditional and cloud-based data centers.

Additional information is available on the RoCE Initiative website including white papers, solution briefs and contact information.

Use of Bookmarks

This document uses a feature known as "bookmarks" that can be used for navigation within this document. Enable the display of bookmarks in your PDF reader so that you can navigate to any section of this document by clicking on the bookmark entry.

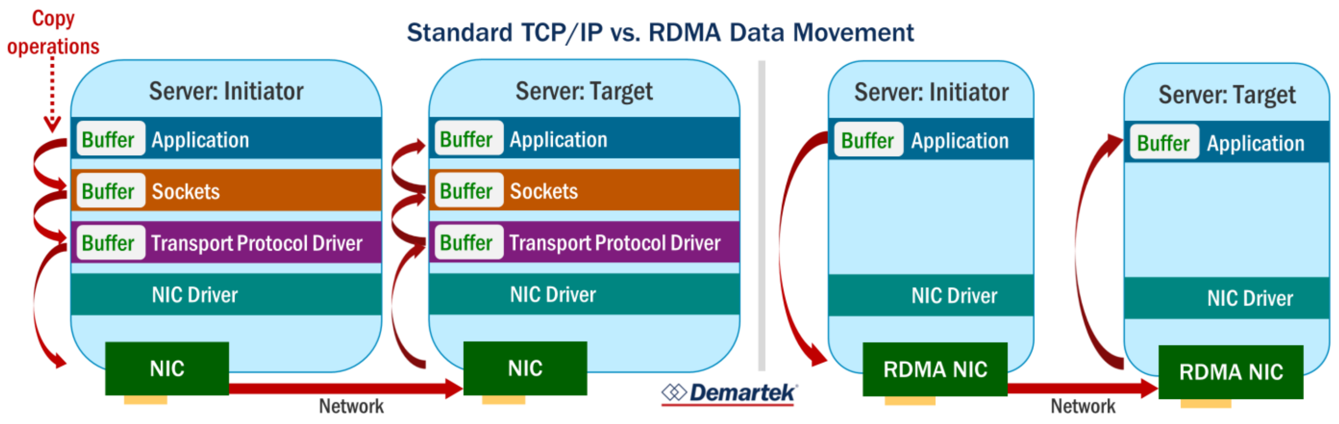
About Demartek

Demartek is a computer industry analyst organization with its own ISO 17025 accredited computer test lab. We provide real-world, hands-on research and analysis by focusing on lab validation testing and power efficiency testing of data center computer equipment such as servers, networking and storage systems. We produce white papers and videos that provide the results of our tests along with our commentary. Additional information regarding our services is available on our lab testing page.

To be notified when new Deployment Guides and lab validation reports become available, you can subscribe to our free newsletter, ***Demartek Lab Notes***, available on our website. We do not give out, rent or sell our email list.

RoCE and RDMA vs. TCP/IP

Remote Direct Memory Access (RDMA) enables more direct movement of data in and out of a server than Transmission Control Protocol/Internet Protocol (TCP/IP). This is displayed in the ***Standard TCP/IP vs. RDMA Data Movement*** diagram below that shows a standard network connection on the left and an RDMA connection on the right.



RDMA bypasses the normal system software network stack components and the multiple buffer copy operations that they normally perform. This elimination of buffer copy operations reduces overall processor consumption and improves (lowers) the latency of the host software stack, since it uses fewer instructions to complete a data transfer.

RDMA can be used in Ethernet applications by using specialized adapters that support RDMA. The adapters are sometimes known as RNICs or RDMA network interface cards (NICs).

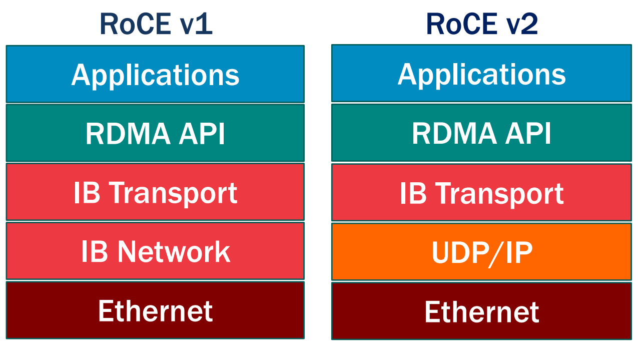
RDMA can be implemented for networking and/or storage applications. Some RDMA adapters may have offload functions built into them. The initiator and the target adapters must use the same type of RDMA technology, such as RoCE v1, RoCE v2, etc.

RDMA can be used for storage traffic such as block protocols like Internet Small Computer Systems Interface (iSCSI) and file protocols such as Network File System (NFS) and Server Message Block (SMB), formerly known as Common Internet File System, or CIFS. RDMA can improve the latency of storage access and reduce processor consumption on the host server performing I/O requests, allowing a higher rate of I/O requests to be sustained, or perhaps a smaller server to perform the same rate of I/O requests.

RoCE can also be used for non-storage applications such as Apache Spark over RDMA (SparkRDMA) and many high-performance computing (HPC) applications that use the RDMA “verbs.”

RoCE v1 vs. RoCE v2

The original implementation of RoCE, known as “v1,” provided the semantics to allow devices to perform DMA transfers that significantly reduce the CPU activity by eliminating the copy functions as shown in the ***Standard TCP/IP vs. RDMA Data Movement*** diagram. RoCE v1 does not span across IP subnets. RoCE v2 enables communication across IP subnets.



A more detailed technical explanation of the difference between RoCE v1 and v2 is available at https://community.mellanox.com/docs/DOC-1451

DCB Switches and Components

To achieve low latency, RoCE adapters work best with Ethernet switches that support Data Center Bridging (DCB) that provide lossless characteristics and support priority flow control (PFC). DCB Ethernet switches have become more common over the last few years and have the following mechanisms available for achieving lossless traffic:

Global Pause/Flow Control – Lossless traffic requires the use of pause and buffers to store in-flight packets until the receiver is ready. The simplest way to implement this is with global pause – if there is too much traffic on a link, a pause will be sent back and all traffic will be stored in a buffer until the receiver is ready.

Pause will happen on links one at a time, propagating back through each hop until information reaches the sender. This causes two problems:

> **Unnecessary Pause:** When using converged ethernet, multiple types of traffic will be sent on the same link, and all traffic types will be paused during congestion, not just the traffic creating the congestion.

> **Congestion Spread**: All stops along a packet path from sender to receiver must be paused before the sender stops sending. It may take a while for the information to propagate through the network. Meanwhile, all intermediary hops

© 2018 Demartek

are paused with buffers full, affecting other traffic that has a different, non-congested final destination but needs to use the link that has been paused.

For the examples in this document, global pause will not be used.

Priority Flow Control (PFC) – PFC assigns each type of traffic a priority index number, usually with an associated virtual LAN (vLAN). This addresses the problem of unnecessary pause, by treating traffic differently by providing flow control based on that traffic's priority index number. Differentiated Services Code Point (DSCP) can also be used to mark packet priorities, with each DSCP mapping to a different PFC. Each type of traffic has a separate buffer on the switch. When the buffer for a lossless priority becomes full due to congestion, that priority is paused, but other priorities may continue. Since one priority can be lossless while others are lossy, the SAN traffic can get paused, allowing the other network traffic to continue or be dropped as necessary. The trigger for pause is a threshold set on the receive buffers.

All of the examples in this document will use PFC and vLAN, as this appears to be the simplest implementation appropriate for Converged Ethernet.

ETS (Enhanced Transmission Selection) – Each traffic class is assigned or guaranteed a percentage of the network bandwidth.

RoCEv2 Congestion Management (RCM) – RCM addresses the congestion spread problem by employing a mechanism to reduce the number of paused links.

> **Data Center Quantized Congestion Notification (DCQCN)** is an example of RCM. The **Explicit Congestion Notification (ECN)** bits are used to mark packets that experienced congestion. If at any point in the packet path congestion is experienced, that switch will mark the packet as having experienced congestion. For this protocol, the triggering threshold for ECN is set on the transmit buffers. When the final receiver, also known as the Notification Point (NP), gets the marked packet, it will send a Congestion Notification Packet (CNP) back to the packet sender or Reaction Point (RP), which will adjust the packet flow. This solves the congestion spread problem as long as a CNP packet does not get dropped. Some vendors advise that CNP be assigned a PFC or DSCP priority that is lossless to avoid this problem, while other vendors do not support this.

> There may be other algorithms developed in future that also address congestion spread, however for the purposes of this guide DCQCN will be used for RCM.

Each example in this document will include optional instructions for adding ETS and DCQCN to the existing PFC setup where available. DSCP and/or a separate priority for CNP are not implemented in our examples.

In all of our examples, RoCE traffic will be on priority 5, VLAN 2, and marked as lossless. All other traffic will be on priority 0 and lossy. While any DCB priority number from 0 to 7 could be used for the RoCE traffic, normally 0 is for the common lossy traffic class (as used here), 3 is hard coded in may switches for FCoE and 4 is commonly used for lossless iSCSI-TLV. ETS, when configured, will allocate 90-95% of bandwidth to RoCE. When DCQCN is used, each endpoint will be configured as both NP and RP.

RoCE Applications

There are a variety of applications that can take advantage of RoCE technology, including the three described below. Typically, RoCE is available for 10Gbps and faster technologies. Increased throughput, reduced CPU consumption, and low latency are among the benefits available.

Windows: SMB Direct and Storage Spaces Direct

Beginning with Windows Server 2012, Microsoft has included a feature known as SMB Direct that supports the use of RDMA-capable network adapters, such as RoCE network adapters. SMB Direct uses SMBv3 (SMB3) and these adapters to enable file transfer operations to operate at high speed and very low latency, improving the overall performance of a file server. SMB Direct is enabled automatically if RDMA-capable adapters are present.

Beginning with Windows Server 2016, Storage Spaces Direct takes advantage of RDMA-capable network adapters to provide highly scalable software-defined storage. Storage Spaces Direct can be deployed on physical machines or virtual machines with Hyper-V and can be deployed in converged or hyper-converged configurations.

VMFleet, a set of automated Powershell scripts for Hyper-V, has been developed to test the capabilities of Hyperconverged Storage Spaces Direct Configurations, using Diskspd to run I/O from each VM.

Linux: iSER and NFS over RDMA

For Linux environments, iSCSI Extensions for RDMA (iSER) allows iSCSI (block storage) traffic to take advantage of RDMA-capable network adapters. Similarly, NFS over RDMA (NFSoRDMA) allows file traffic to take advantage of RDMA-capable network adapters.

VMware: iSER over RDMA and RDMA for Guest VMs

Starting with vSphere 6.5, VMware introduced RoCE mode. Drivers are now available to allow the creation of an iSER storage adapter in the ESXi host, or to allow guest VMs to directly use the RoCE abilities of the ethernet adapters via SR-IOV or passthrough.

NVMe over Fabrics

NVM Express over Fabrics (NVMe-oF™) defines a common architecture that supports a range of storage networking fabrics for NVMe block storage protocol, including RDMA fabrics. RoCE is supported for NVMe-oF.

Products Tested in this Guide

We used several different products in various configurations to produce this ***RoCE Deployment Guide***. These include the following:

- HPE 5900 Series JG838A 10/40GbE switch
- Mellanox® SN2410 25GbE/100GbE switch
- Mellanox ConnectX®-5 MCX512A 2x25GbE and ConnectX-5 MCX515A 100GbE adapters
- Cavium® FastLinQ™ QL45611HLCU-CK 1x100GbE/4x25GbE and QL41234HLCU 4x25GbE adapters
- Broadcom® NetXtreme® BCM957414A4142CC 2x25GbE adapters

Our previous deployment guide used some older adapters and storage and remains available on the Demartek website.

HPE

The HPE 5900 Series JG838A is a high-density, ultra-low-latency, top of rack switch with 48-ports of 10GbE and 4 ports of 40GbE.

Mellanox

The Mellanox SN2410 is an Open Ethernet top-of-rack switch with 48-ports of 25GbE and 8-ports of 100GbE.

The Mellanox ConnectX-5 MCX512A supports 1/10/25 GbE. The Mellanox ConnectX-5 MCX515A supports 10/50/100GbE.

Mellanox provides recommended RoCE configuration examples on their website.

Cavium

The Cavium FastLinQ 41000 Series Ethernet Adapters support up to 4 ports of 10/25GbE. The Cavium 45000 Series Ethernet Adapters support 10/25/40/50/100GbE.

Cavium provides RoCE and related resources on their Universal RDMA product page.

On July 6, 2018, Marvell Technology Group Ltd. announced the completion of its acquisition of Cavium, Inc. This acquisition was first announced on November 20, 2017.

Due to the very recent completion of this acquisition, references in this document to Cavium components retain this brand name.

Broadcom

The Broadcom NetXtreme BCM957414A4142CC 2x25GbE adapters support 10/20/25/40/50 GbE. Broadcom NetXtreme also carries a 100GbE adapter that was not used in this guide.

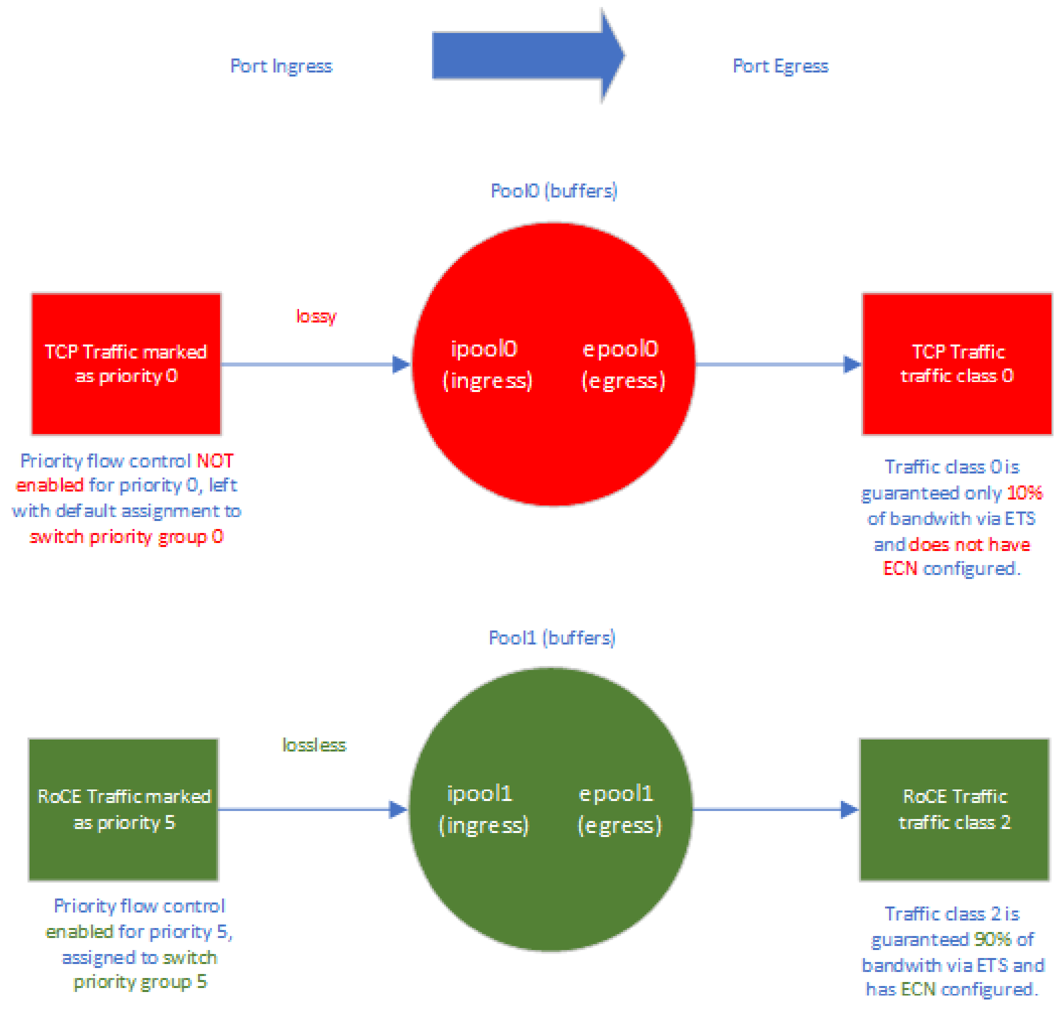
Deploying RoCE on a Network Switch

The user will need to appropriately configure the switch and adapters according to vendor-specific instructions, and then follow the appropriate application setup.

The user should keep in mind that usually jumbo frames are not necessary in a RoCE configuration.

Deploying RoCE on a Mellanox Switch

A typical way to achieve this setup on a Mellanox switch is as follows:


Port Ingress
Port Egress
Pool0 (buffers)
TCP Traffic marked as priority 0
lossy
ipool0 (ingress)
epool0 (egress)
TCP Traffic traffic class 0
Priority flow control NOT enabled for priority 0, left with default assignment to switch priority group 0
Traffic class 0 is guaranteed only 10% of bandwith via ETS and does not have ECN configured.
Pool1 (buffers)
RoCE Traffic marked as priority 5
lossless
ipool1 (ingress)
epool1 (egress)
RoCE Traffic traffic class 2
Priority flow control enabled for priority 5, assigned to switch priority group 5
Traffic class 2 is guaranteed 90% of bandwith via ETS and has ECN configured.

(Default username and password for a new Mellanox switch is admin,admin).

1. Gain access to the administrator command set:
   ]> enable
   ]# config t
2. Display the current configuration:
   ](config)# show running-config
3. Display an interface:
   ](config)# show interfaces ethernet 1/1
4. Make sure flow control is disabled (often it is best to revert to the initial configuration):
   A list of saved configurations may be obtained by using a ? instead of “initial” in the command below:
   ](config)# config switch-to initial
5. Enable PFC priority 5 on the switch:

](config)# dcb priority-flow-control enable force
](config)# dcb priority-flow-control priority 5 enable

6. Enable PFC on **each port**:
   ](config)# interface ethernet 1/1 dcb priority-flow-control mode on force
   ](config)# interface ethernet 1/2 dcb priority-flow-control mode on force
   …
7. Configure VLAN on switch:
   ](config)# vlan 2
   ](config)# exit
8. Configure buffer pools on switch. (Each pool contains an ingress pool (iPool) and an egress pool (ePool)).
9. **Configure a small buffer Pool 0 for TCP/IP:**
   ](config)# pool ePool0 size 956051 type dynamic
   ](config)# pool iPool0 size 956051 type dynamic
   Configure a Larger buffer Pool 1 for RoCE:
   ](config)# pool iPool1 size 8604460 type dynamic
   ](config)# pool ePool1 size 16777000 type dynamic
10. Configure VLAN on **each port**:
    ](config)# interface ethernet 1/1
    ](config interface ethernet 1/1)# switchport mode trunk
    ](config interface ethernet 1/1)# switchport trunk allowed-vlan 2
    ](config interface ethernet 1/1)# exit
    …and repeat for next port:
    ](config)# interface ethernet 1/2
    ](config interface ethernet 1/2)# switchport mode trunk
    ](config interface ethernet 1/2)# switchport trunk allowed-vlan 2
    …
11. Configure lossless (RoCE) and lossy (all other traffic) buffers for **each port** by doing the following:
    -](config)# interface ethernet 1/1
    - Map RoCE switch-priority 5 to its own priority group (pg). Right now all switch-priorities are mapped by default to priority group 0 (iPort.pg0). We will keep things simple and use priority group 5 for switch-priority 5, but any priority group other than 0 could be used.

      ](config interface ethernet 1/1)# ingress-buffer iPort.pg5 bind switch-priority 5
    - Configure traffic classes (tc) for the switch-priorities. Traffic class 0 will be used for priority 0, and traffic class 2 will be used for priority 5.:

      ](config interface ethernet 1/1)# traffic-class 0 bind switch-priority 0
      ](config interface ethernet 1/1)# traffic-class 2 bind switch-priority 5
    - Map the corresponding priority groups and traffic classes to the same buffer pool.

      First TCP/IP:
      ](config interface ethernet 1/1)# ingress-buffer iPort.pg0 map pool iPool0 type lossy reserved 20480 shared alpha 8
      ](config interface ethernet 1/1)# egress-buffer ePort.tc0 map pool ePool0 reserved 1500 shared alpha 2

      Then RoCE:
      ](config interface ethernet 1/1)# ingress-buffer iPort.pg5 map pool iPool1 type lossless reserved 34816 xoff 8192 xon 8192 shared alpha 2
      ](config interface ethernet 1/1)# egress-buffer ePort.tc2 map pool ePool1 reserved 1500 shared alpha inf

      ](config interface ethernet 1/1)# exit

    …and repeat for the next port. Each port gets five to seven commands total to configure the buffers, depending on whether ETS/ECN is to be configured later.
12. *Optional:* Configure ETS and ECN **on each port:**

```
](config)# interface ethernet 1/1
](config interface ethernet 1/1)# traffic-class 2
congestion-control ecn minimum-absolute 150
maximum-absolute 1500
](config interface ethernet 1/1)# traffic-class 0
dcb ets wrr 10
](config interface ethernet 1/1)# traffic-class 1
dcb ets wrr 0
](config interface ethernet 1/1)# traffic-class 2
dcb ets wrr 90
](config interface ethernet 1/1)# traffic-class 3
dcb ets wrr 0
](config interface ethernet 1/1)# traffic-class 4
dcb ets wrr 0
](config interface ethernet 1/1)# traffic-class 5
dcb ets wrr 0
](config interface ethernet 1/1)# traffic-class 6
dcb ets wrr 0
](config interface ethernet 1/1)# traffic-class 7
dcb ets wrr 0
](config interface ethernet 1/1)# exit
```

…repeat for next port

```
](config)# interface ethernet 1/2 …
```

13. Write the configuration to come back to later:

```
](config)# config write to RoCEconfig
```

14. Switch back to it at any time if you make changes:

```
](config)# config switch-to RoCEconfig
```

15. Double-check with the following commands:

```
](config)# show dcb priority-flow-control
](config)# show interfaces ethernet 1/1 counters
](config)# show interfaces ethernet 1/1
congestion-control
](config)# show buffer status interfaces ethernet
1/1
](config)# show interfaces ethernet 1/1 counters
priority 5
```

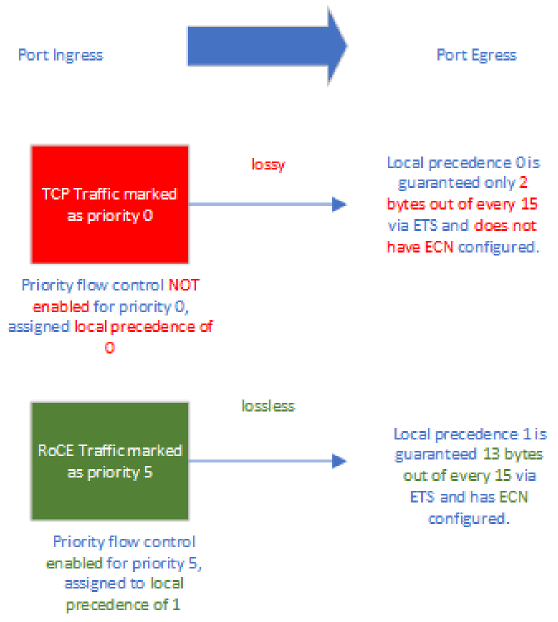
For ETS configurations only:

](config)# show dcb ets interface ethernet 1/1

Most of the time, this configuration change will require a reboot. If PFC is not enabled but all configuration changes are complete, a manual reboot may be needed after the configuration is written.

Deploying RoCE on a HPE 5900 Series Switch

A typical way to achieve this setup on a HPE 5900 Series switch is as follows:


Port Ingress
Port Egress
TCP Traffic marked as priority 0
lossy
Local precedence 0 is guaranteed only 2 bytes out of every 15 via ETS and does not have ECN configured.
Priority flow control NOT enabled for priority 0, assigned local precedence of 0
RoCE Traffic marked as priority 5
lossless
Local precedence 1 is guaranteed 13 bytes out of every 15 via ETS and has ECN configured.
Priority flow control enabled for priority 5, assigned to local precedence of 1

1. Gain access to the administrator command set:
   <HPE5900> enable

2. Display the current configuration:
   [HPE5900] display current-configuration

3. Display an interface:
   [HPE5900] display interface Ten-GigabitEthernet 1/0/1
   [HPE5900] display interface FortyGigE 1/0/49

4. Make sure flow control is disabled (often it is best to revert to the initial configuration):
   First get a list of save configurations:
   [HPE5900] exit
   <HPE5900> dir
   Now revert to the initial configuration:
   <HPE5900> system-view
   [HPE5900] configuration replace file initialconfig.cfg

   *Note that the switch will not automatically enter the selected configuration upon reboot without further configuration.

5. Configure VLAN on switch:
   [HPE5900] vlan 2
   [HPE5900-vlan2] quit

6. Enable PFC and VLAN on **each port**:
   [HPE5900] interface Ten-GigabitEthernet 1/0/1
   [HPE5900-Ten-GigabitEthernet1/0/1] port link-type trunk
   [HPE5900-Ten-GigabitEthernet1/0/1] port trunk permit vlan 2
   [HPE5900-Ten-GigabitEthernet1/0/1] priority-flow-control auto
   [HPE5900-Ten-GigabitEthernet1/0/1] priority-flow-control no-drop dot1p 5
   [HPE5900-Ten-GigabitEthernet1/0/1] qos trust dot1p
   [HPE5900-Ten-GigabitEthernet1/0/1] quit

   Repeat …
   [HPE5900] interface Ten-GigabitEthernet 1/0/2
   …

16. *Optional*: Configure ETS and ECN
   Map 802.1p-priority to local-precedence
   [HPE5900] qos map-table dot1p-lp
   [HPE5900-maptbl-dot1p-lp] import 0 export 0
   [HPE5900-maptbl-dot1p-lp] import 1 export 0
   [HPE5900-maptbl-dot1p-lp] import 2 export 0
   [HPE5900-maptbl-dot1p-lp] import 3 export 0
   [HPE5900-maptbl-dot1p-lp] import 4 export 0
   [HPE5900-maptbl-dot1p-lp] import 5 export 1
   [HPE5900-maptbl-dot1p-lp] import 6 export 0
   [HPE5900-maptbl-dot1p-lp] import 7 export 0
   [HPE5900-maptbl-dot1p-lp] quit
   [HPE5900] qos wred queue table Table1
   [HPE5900] queue 5 low-limit 150 high-limit 1500
   [HPE5900] queue 5 ecn

7. *Optional:* Configure ETS and ECN **on each port:**
   [HPE5900] interface Ten-GigabitEthernet 1/0/1
   [HPE5900-Ten-GigabitEthernet1/0/1] qos wrr 0 group 1 byte-count 2
   [HPE5900-Ten-GigabitEthernet1/0/1] qos wrr 1 group 1 byte-count 13
   [HPE5900-Ten-GigabitEthernet1/0/1] qos wrr 2 group sp
   [HPE5900-Ten-GigabitEthernet1/0/1] qos wrr 3 group sp
   [HPE5900-Ten-GigabitEthernet1/0/1] qos wrr 4 group sp
   [HPE5900-Ten-GigabitEthernet1/0/1] qos wrr 5 group sp
   [HPE5900-Ten-GigabitEthernet1/0/1] qos wrr 6 group sp
   [HPE5900-Ten-GigabitEthernet1/0/1] qos wrr 7 group sp
   [HPE5900-Ten-GigabitEthernet1/0/1] qos trust dot1p
   [HPE5900-Ten-GigabitEthernet1/0/1] qos wred apply Table1
   [HPE5900-Ten-GigabitEthernet1/0/1] quit
8. Write the configuration to come back to later:
   [HPE5900] save RoCEConfiguration.cfg

9. Switch back to it at any time if you make changes:
   [HPE5900] configuration replace file RoCEConfiguration.cfg

10. Double-check with the following commands:

[HPE5900] display interface brief
[HPE5900] display vlan 2
[HPE5900] display current-configuration
For ETS and ECN Configurations Only:
[HPE5900] display qos map-table dot1p-lp
[HPE5900] display qos wred interface Ten-GigabitEthernet 1/0/1
[HPE5900] display qos wred table name Table1

Windows: Deploying SMB Direct over RoCE with Storage Spaces

If a Storage Spaces Direct Cluster is desired, the user will need to install Windows Server 2016 Datacenter Edition on all cluster nodes. For SMB Direct, Windows Server 2012 or higher is required.

For all configurations, the user will need to appropriately configure the switch, configure the adapters according to vendor-specific instructions, and then follow the appropriate application setup. The reader should also configure RSS for optimal distribution of queues between processors.

TCP/IP connection setup should be completed before proceeding to the adapter and application setup steps. The steps to do so are not outlined here in this guide.

Adapter and application setup will need to be completed on each server in a configuration.

Mellanox and Broadcom

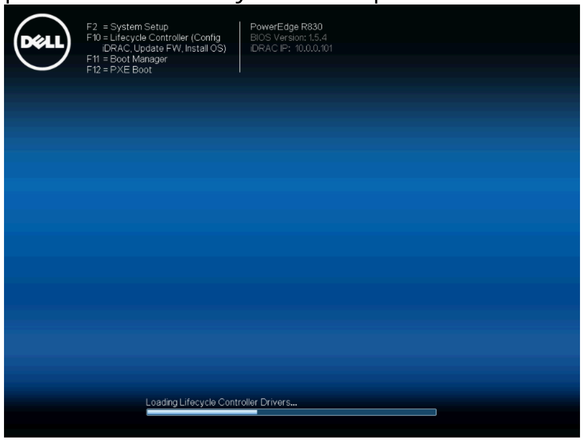
To enable PFC and QoS within the Windows Server 2016 OS, the Data Center Bridging Feature is installed using the Add Roles and Features Wizard. This will enable the following PowerShell commands which put RoCE on PFC Priority 5:

1. Set up QoS:
   PS:\>Remove-NetQosTrafficClass
   PS:\>Remove-NetQosPolicy -Confirm:$False
   PS:\> Set-NetQosDcbxSetting -Willing $false
   PS:\> New-NetQosPolicy "SMB" -SMB -PriorityValue8021Action 5
   PS:\> New-NetQosPolicy "DEFAULT" -Default -PriorityValue8021Action 0
   PS:\>Disable-NetQosFlowControl 0,1,2,3,4, 6,7
   PS:\>Enable-NetQosFlowControl -Priority 5
2. Enable QoS on your adapter:
   First find your adapter:
   PS:\> Get-NetAdapter
   Then enable QoS on your adapter:
   PS:\>Enable-NetAdapterQos -InterfaceAlias "Ethernet 5"
3. *Optional: Enable ETS*
   PS:\> New-NetQosTrafficClass "SMB" -Priority 5 -BandwidthPercentage 95 -Algorithm ETS
4. *Optional (Mellanox Only): Enable ECN*
   PS:\> MLx5Cmd.exe -Qosconfig -Name "Ethernet 5" -Dcqcn -Enable 5

Cavium

In order to use RoCE with Cavium cards, the DCBX and RoCE priority are selected in the UEFI HII interface. Here is an example of this on a Dell server:

1. When the menu appears at boot in the top right, press F2 to enter System Setup.

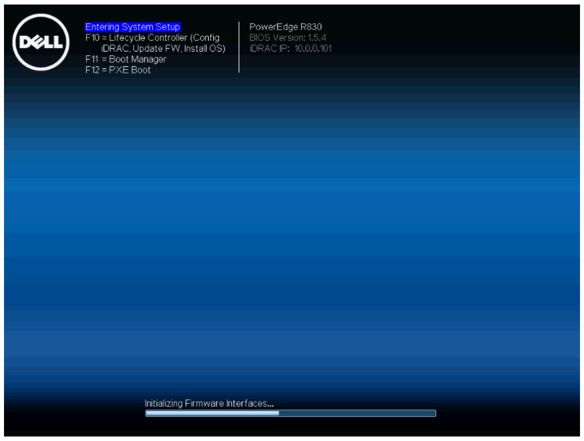


F2 = System Setup
F10 = Lifecycle Controller (Config iDRAC, Update FW, Install OS)
F11 = Boot Manager
F12 = PXE Boot

PowerEdge R830
BIOS Version: 1.5.4
iDRAC IP: 10.0.0.101

Loading Lifecycle Controller Drivers...

2. Wait while normal boot procedures are completed.

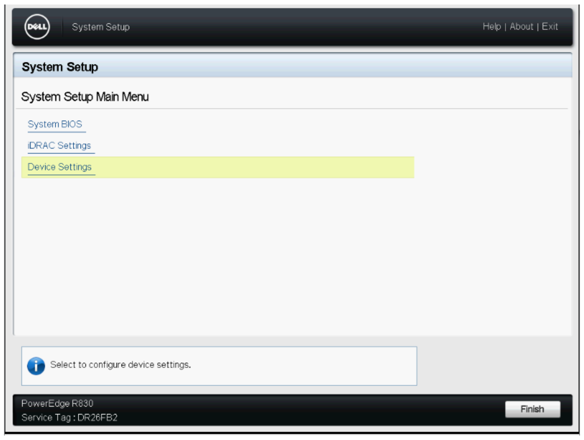


Entering System Setup
F10 = Lifecycle Controller (Config iDRAC, Update FW, Install OS)
F11 = Boot Manager
F12 = PXE Boot

PowerEdge R830
BIOS Version: 1.5.4
iDRAC IP: 10.0.0.101

Initializing Firmware Interfaces...

3. Select Device Settings at the System Setup Main Menu.



System Setup

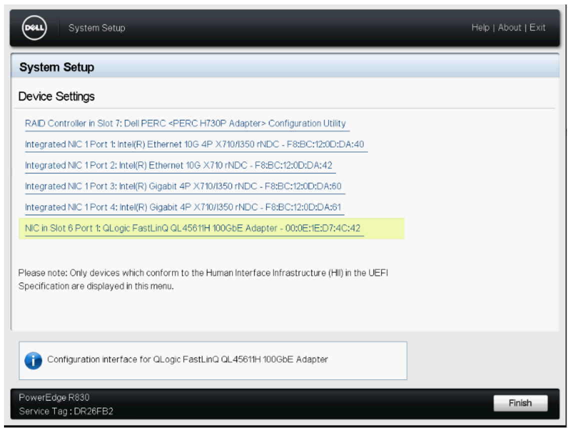
System Setup Main Menu

System BIOS
iDRAC Settings
Device Settings

Select to configure device settings.

PowerEdge R830
Service Tag : DR26FB2

4. Choose the Cavium Device to configure.



System Setup

Device Settings

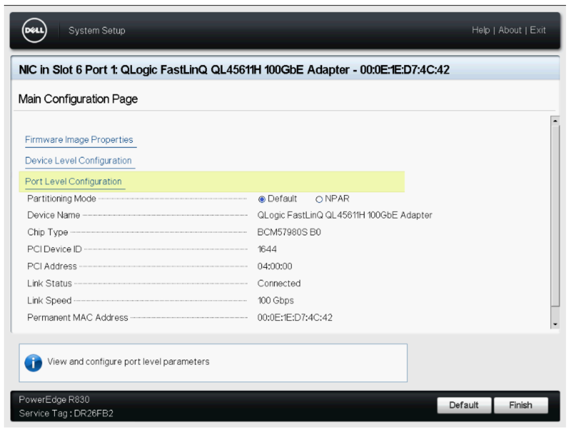
RAID Controller in Slot 7: Dell PERC <PERC H730P Adapter> Configuration Utility
Integrated NIC 1 Port 1: Intel(R) Ethernet 10G 4P X710/I350 rNDC - F8:BC:12:0D:DA:40
Integrated NIC 1 Port 2: Intel(R) Ethernet 10G X710 rNDC - F8:BC:12:0D:DA:42
Integrated NIC 1 Port 3: Intel(R) Gigabit 4P X710/I350 rNDC - F8:BC:12:0D:DA:60
Integrated NIC 1 Port 4: Intel(R) Gigabit 4P X710/I350 rNDC - F8:BC:12:0D:DA:61
NIC in Slot 6 Port 1: QLogic FastLinQ QL45611H 100GbE Adapter - 00:0E:1E:D7:4C:42

Please note: Only devices which conform to the Human Interface Infrastructure (HII) in the UEFI Specification are displayed in this menu.

Configuration interface for QLogic FastLinQ QL45611H 100GbE Adapter

PowerEdge R830
Service Tag : DR26FB2

5. Select Port Level Configuration.



NIC in Slot 6 Port 1: QLogic FastLinQ QL45611H 100GbE Adapter - 00:0E:1E:D7:4C:42

Main Configuration Page

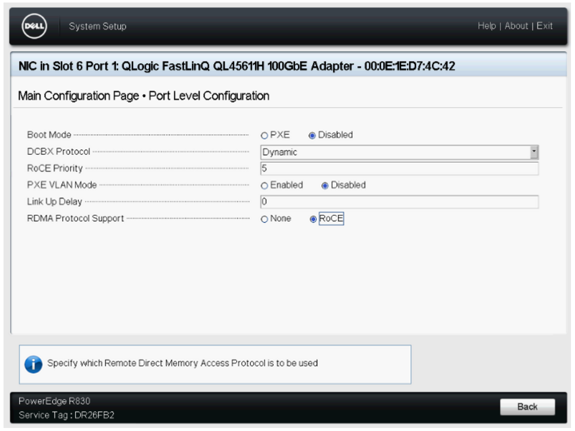
Firmware Image Properties
Device Level Configuration
Port Level Configuration

Partitioning Mode	[✓ Default] NPAR
Device Name	QLogic FastLinQ QL45611H 100GbE Adapter
Chip Type	BCM57980S B0
PCI Device ID	1644
PCI Address	04:00:00
Link Status	Connected
Link Speed	100 Gbps
Permanent MAC Address	00:0E:1E:D7:4C:42

View and configure port level parameters

PowerEdge R830
Service Tag : DR26FB2

6. Select Dynamic DCBX protocol, RoCE for RDMA Protocol support, and choose an appropriate RoCE Priority number.



NIC in Slot 6 Port 1: QLogic FastLinQ QL45611H 100GbE Adapter - 00:0E:1E:D7:4C:42

Main Configuration Page • Port Level Configuration

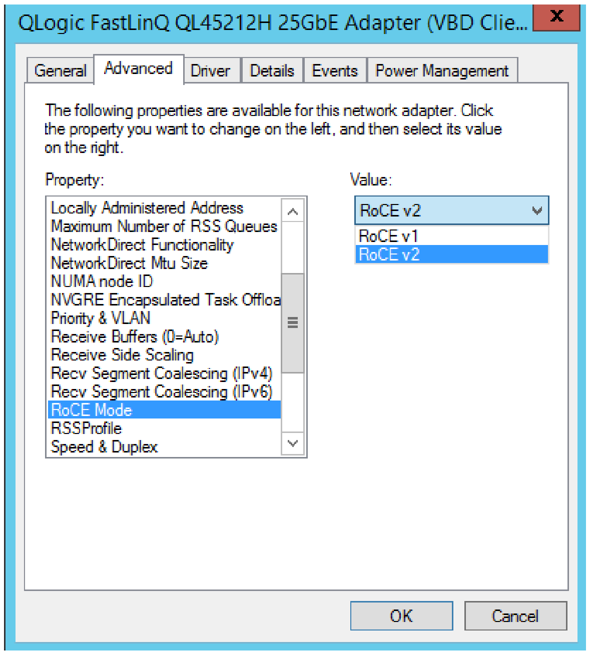
Boot Mode	PXE [✓ Disabled]
DCBX Protocol	Dynamic
RoCE Priority	5
PXE VLAN Mode	Enabled [✓ Disabled]
Link Up Delay	0
RDMA Protocol Support	None [✓ RoCE]

Specify which Remote Direct Memory Access Protocol is to be used

PowerEdge R830
Service Tag : DR26FB2

Other server manufacturers should have similar UEFI hook-ins for the Cavium cards allowing the DCBX option and RoCE priority class to be set. If any difficulty is encountered using the HII interface on a server, contact Cavium technical support for an engineering solution.

7. Upon boot, there is the option to specify the RoCE mode as RoCE v2 or RoCE v1. RoCE v2 was used for all our SMB Direct testing.


QLogic FastLinQ QL45212H 25GbE Adapter (VBD Clie...

General | Advanced | Driver | Details | Events | Power Management

The following properties are available for this network adapter. Click the property you want to change on the left, and then select its value on the right.

Property:
Locally Administered Address
Maximum Number of RSS Queues
NetworkDirect Functionality
NetworkDirect Mtu Size
NUMA node ID
NVGRE Encapsulated Task Offloa
Priority & VLAN
Receive Buffers (0=Auto)
Receive Side Scaling
Recv Segment Coalescing (IPv4)
Recv Segment Coalescing (IPv6)
RoCE Mode
RSSProfile
Speed & Duplex

Value:
RoCE v2
RoCE v1
RoCE v2

8. Network Direct MTU size for Cavium may be increased to 4096 for possible performance gains, as long as the Ethernet MTU size is larger than 4096, and a 4096 MTU is configured on the switch, client, and server.

SMB Direct Setup

When deploying SMB Direct over RoCE, perform the following steps after switch configuration, OS install, and adapter configuration:

1. Make sure the global setting for SMB Direct is enabled using the PowerShell command:
   \>get-NetOffloadGlobalSetting
2. Make sure multichannel is enabled on the Client side.
   \> get-SMBClientConfiguration | Select EnableMultichannel
   OR on the Server side
   \> get-SMBServerConfiguration | Select EnableMultichannel
3. Verify the network interface is listed as being RDMA capable on the Client side:
   \>get-SMBClientNetworkInterface
   OR on the Server side
   \>get-SMBServerNetworkInterface
   AND
   \>get-NetAdapterRDMA
4. If all is configured correctly, there should be two SMB Direct Listeners waiting for connections:
   \>netstat –xan 1
   Once a connection to the remote file share is initiated and traffic has started, netstat will show the connections.
5. To view RDMA connection attempts, active connections, bytes sent, or frames sent, open the Perfmon program, right click to add counters, and select RDMA.
6. While not employed in our own testing, the following may be explored to potentially enhance performance of SMB Direct:

   > Increase the number of SMB Direct RDMA connections from the default (2) to 8 or 16 using the PowerShell command

   \>Set-ItemProperty –Path "HKLM:\SYSTEM\CurrentControlSet\Services\LanmanWorkstation\Parameters" ConnectionCountPerRdmaNetworkInterface –Type DWORD –Value <number of connections> -Force
   It should be noted that there will not be a key present in the registry if the default number of connections are being used; it will only be present once the value is changed. It should also be noted that the PowerShell command sets the value immediately, while using regedit necessitates a reboot. This option will have to be set on both the target server and on the client.

   > If RoCE v2 is not necessary for routing purposes, RoCE v1 may give better performance.

Once SMB Direct has been validated, it can be used to create a Storage Spaces Direct cluster or some Generic File Sharing.

Application Setup – Storage Spaces Direct

A Storage Spaces Direct (S2D) Cluster may be utilized to create a Scale-Out (always available) file server or to create a Hyperconverged Infrastructure.

1. Install the desired block devices locally in the cluster nodes. Follow standard Microsoft guidelines for allowed storage configurations. The simplest storage configuration to deploy is four NVMe devices per node. If SAS devices are to be used, SAS Enclosure Services must be available, a Host Bus Adapter (HBA) must be used instead of a RAID controller, and a caching tier must be provided.
2. Add the cluster nodes to the domain.
3. For redundancy, a minimum of two identical network ports should be teamed together for use by Hyper-V. However, a single port can be used to create a vSwitch alone if a second port is not available. Traditional Load Balancer Fail Over (LBFO) teaming is not compatible with RDMA. Instead, rdma-capable ports should be teamed in a vSwitch using Switch Embedded Teaming (SET). If all network adapters included in a SET vSwitch are RDMA capable, the resultant vSwitch will be RDMA capable as well:

   PS:\> Get-NetAdapter

   Find out the adapter names from the above command. In our example, we use “Ethernet 3” and “Ethernet 5.”

   PS:\>New-VMSwitch –Name SETswitch –NetAdapterName “Ethernet 3”,”Ethernet 5” –EnableEmbeddedTeaming $true –AllowManagementOS $false

   PS:\> Add-VMNetworkAdapter –SwitchName SETswitch –Name SMB_1 –managementOS

   PS:\> Add-VMNetworkAdapter –SwitchName SETswitch –Name SMB_2 –managementOS

   PS:\> Set-VMNetworkAdapterTeamMapping –ManagementOS –SwitchName SETswitch –VMNetworkAdapterName “SMB_1” –PhysicalNetAdapterName “Ethernet 3”

   PS:\> Set-VMNetworkAdapterTeamMapping –ManagementOS –SwitchName SETswitch –VMNetworkAdapterName “SMB_2” –PhysicalNetAdapterName “Ethernet 5”

   PS:\> Set-VMNetworkAdapterVlan –VMNetworkAdapterName SMB_1 –VlanId 2 –Access –ManagementOS

   PS:\> Set-VMNetworkAdapterVlan –VMNetworkAdapterName SMB_2 –VlanID 2 –Access -ManagementOS

   PS:\> Enable-NetAdapterRDMA “vEthernet (SMB_1)”,”vEthernet (SMB_2) “

   Validate SMB Direct for vEthernet (SMB_1) and vEthernet (SMB_2) before continuing.

4. Create the S2D Cluster. We created a 2-node cluster with mirroring:

   PS:\>Test-Cluster –Node “S2D-Node1”,”S2D-Node2” –Include “Storage Spaces Direct”, “Inventory”, “Network”,”System Configuration”

   PS:\>New-Cluster –Name “My-S2DCluster” –Node “S2D-Node1”,”S2D-Node2” –NoStorage –StaticAddress 10.0.2.55

   As we have a 2-node cluster, we had to configure the witness.

   We also erased all partitions on our NVMe so that all NVMe storage was presented as primordial in Storage Spaces and then added the storage to S2D.

   PS:\> Enable-ClusterStorageSpacesDirect –CimSession “My-S2DCluster”

5. In our example, we configured virtual disks in our S2D Storage Pool for our VMFleet VMs to use in a Hyperconverged configuration.

Application Setup – Generic File Server

Install the desired block device(s) locally in the target server and use Storage Spaces to create Virtual Drives containing File Shares.

Demartek RoCE Deployment Guide 2018 Edition

Linux: Deploying iSER or NFS over RoCE

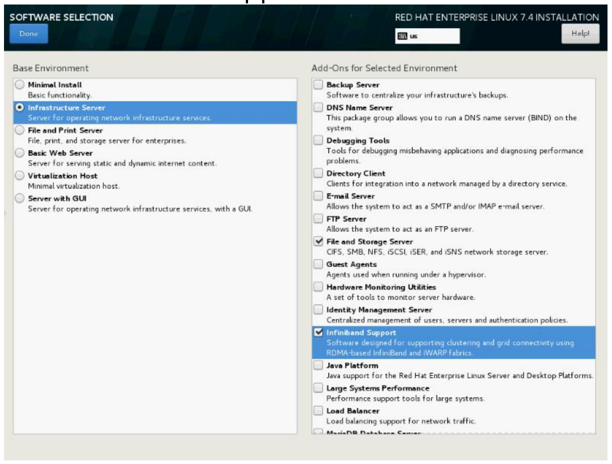
The user will need to appropriately configure the switch, configure the adapters according to vendor-specific instructions, and then follow the appropriate application setup.

TCP/IP connection setup should be completed before proceeding to adapter and application setup steps. The steps to do so are not outlined here in this guide.

Adapter and application setup will need to be completed for each server.

Mellanox

1. Install an InfiniBand OFED. There are two different recommendations based on application choice.
   > For iSER, Download the Mellanox OFED. (This will also install any needed firmware updates).
     > Install prerequisite packages for your OS as listed in MLNx-X_OFED_README:
       # yum install perl pciutils python gcc-gfortran libxml2-python tcsh libnl.i686 libnl expat glib2 tcl libstdc__ bc tk g tk2 atk cairo numactl pkgconfig ethtool
       > Install the Mellanox OFED
       #./mnt/mlnxofedinstall
       > Load the new driver
       #/etc/init.d/openibd restart
   > For NFSoRDMA, obtain the inbox OFED. This can be installed at OS installation by checking off "Infiniband Support."



Base Environment

- ( ) **Minimal Install** Basic functionality.
- (●) **Infrastructure Server** Server for operating network infrastructure services.
- ( ) **File and Print Server** File, print, and storage server for enterprises.
- ( ) **Basic Web Server** Server for serving static and dynamic internet content.
- ( ) **Virtualization Host** Minimal virtualization host.
- ( ) **Server with GUI** Server for operating network infrastructure services, with a GUI.

Add-Ons for Selected Environment

- [ ] **Backup Server** Software to centralize your infrastructure's backups.
- [ ] **DNS Name Server** This package group allows you to run a DNS name server (BIND) on the system.
- [ ] **Debugging Tools** Tools for debugging misbehaving applications and diagnosing performance problems.
- [ ] **Directory Client** Clients for integration into a network managed by a directory service.
- [ ] **E-mail Server** Allows the system to act as a SMTP and/or IMAP e-mail server.
- [ ] **FTP Server** Allows the system to act as an FTP server.
- [✓] **File and Storage Server** CIFS, SMB, NFS, iSCSI, iSER, and iSNS network storage server.
- [ ] **Guest Agents** Agents used when running under a hypervisor.
- [ ] **Hardware Monitoring Utilities** A set of tools to monitor server hardware.
- [ ] **Identity Management Server** Centralized management of users, servers and authentication policies.
- [✓] **Infiniband Support** Software designed for supporting clustering and grid connectivity using RDMA-based InfiniBand and iWARP fabrics.
- [ ] **Java Platform** Java support for the Red Hat Enterprise Linux Server and Desktop Platforms.
- [ ] **Large Systems Performance** Performance support tools for large systems.
- [ ] **Load Balancer** Load balancing support for network traffic.
- [ ] **MariaDB Database Server**

If the OFED was not installed at OS installation, it can be downloaded and installed later.
  # yum –y groupinstall "InfiniBand Support"

2. Check if the drivers are loaded correctly.
   #ibdev2netdev
   If the drivers are loaded correctly, this should list the name of the Mellanox interface. For example:

```
mlx5_0 port 1 ==> ens15f0 (Up)
mlx5_1 port 1 ==> ens15f1 (Up)
```

   If nothing was present:
     #/etc/init.d/openibd restart
     (It may be necessary to unload several other modules in order to run this command:
       #rmmod ib_isert
       #rmmod xprtrdma
       #rmmod ib_srpt
       #/etc/init.d/openibd restart
       #ibdev2netdev)

3. Configure PFC.
   > Display the current priorities. This should show that there is no priority set.
     #mlnx_qos -i <interface name>

```
PFC configuration:
priority    0   1   2   3   4   5   6   7
enabled     0   0   0   0   0   0   0   0
```

   > Set the desired priority. Each number corresponds to a priority, 0 through 7. Setting the 6th number to 1 enables the 6th priority in the sequence, which is 5 (0,1,2,3,4,5,6,7).
     #mlnx_qos -i <interface name> -f 0,0,0,0,0,1,0,0

   > Set the egress priority mappings. There is supposed to be a way to set the egress mapping by adding a line to the network script. However, in our case it didn't work and we used alternate means. Unfortunately, this set of commands will have to be re-entered each time the server or

network service is restarted. It may be useful to put them in a script.
#ip link set <VLAN interface> type vlan egress
0:<RoCE PFC priority>
#ip link set <VLAN interface> type vlan egress
1:<RoCE PFC priority>
… repeat for all priorities up to 15.

```
#ip link set ens15f0.2 type vlan egress 0:5
#ip link set ens15f0.2 type vlan egress 1:5
#ip link set ens15f0.2 type vlan egress 2:5
#ip link set ens15f0.2 type vlan egress 3:5
#ip link set ens15f0.2 type vlan egress 4:5
#ip link set ens15f0.2 type vlan egress 5:5
#ip link set ens15f0.2 type vlan egress 6:5
#ip link set ens15f0.2 type vlan egress 7:5
#ip link set ens15f0.2 type vlan egress 8:5
#ip link set ens15f0.2 type vlan egress 9:5
#ip link set ens15f0.2 type vlan egress 10:5
#ip link set ens15f0.2 type vlan egress 11:5
#ip link set ens15f0.2 type vlan egress 12:5
#ip link set ens15f0.2 type vlan egress 13:5
#ip link set ens15f0.2 type vlan egress 14:5
#ip link set ens15f0.2 type vlan egress 15:5
#ip link set ens15f1.2 type vlan egress 0:5
#ip link set ens15f1.2 type vlan egress 1:5
#ip link set ens15f1.2 type vlan egress 2:5
#ip link set ens15f1.2 type vlan egress 3:5
#ip link set ens15f1.2 type vlan egress 4:5
#ip link set ens15f1.2 type vlan egress 5:5
#ip link set ens15f1.2 type vlan egress 6:5
#ip link set ens15f1.2 type vlan egress 7:5
#ip link set ens15f1.2 type vlan egress 8:5
#ip link set ens15f1.2 type vlan egress 9:5
#ip link set ens15f1.2 type vlan egress 10:5
#ip link set ens15f1.2 type vlan egress 11:5
#ip link set ens15f1.2 type vlan egress 12:5
#ip link set ens15f1.2 type vlan egress 13:5
#ip link set ens15f1.2 type vlan egress 14:5
#ip link set ens15f1.2 type vlan egress 15:5
```

> Check that the egress commands worked.

```
#cat /proc/net/vlan/ens15f0.2

ens15f0.2  VID: 2 REORDER_HDR: 1  dev->priv_flags: 1
         total frames received
1964
          total bytes received
134626
      Broadcast/Multicast Rcvd
0

      total frames transmitted
1047
       total bytes transmitted
138843
Device: ens15f0
INGRESS priority mappings: 0:0  1:0  2:0
3:0  4:0  5:0  6:0 7:0
 EGRESS priority mappings: 0:5 1:5 2:5 3:5
4:5 5:5 6:5 7:5 8:5 9:5 10:5 11:5 12:5
13:5 14:5 15:5
```

> *Optional*: Check the GIDs with the Mellanox provided script show-gids (below). This script can be obtained from the Mellanox website: https://community.mellanox.com/docs/DOC-2421

```
#./show-gids
DEV             PORT    INDEX   GID                                       IPv4        VER     DEV
---             ----    -----   ---                                    ---------      ---     ---
mlx5_0          1       0       fe80:0000:0000:0000:ee0d:9aff:fe48:aefe               v1      ens15f0
mlx5_0          1       1       fe80:0000:0000:0000:ee0d:9aff:fe48:aefe               v2      ens15f0
mlx5_0          1       2       fe80:0000:0000:0000:f3f5:5f75:d952:f464               v1      ens15f0
mlx5_0          1       3       fe80:0000:0000:0000:f3f5:5f75:d952:f464               v2      ens15f0
mlx5_0          1       4       0000:0000:0000:0000:0000:ffff:0a00:021f  10.0.2.31    v1      ens15f0.2
mlx5_0          1       5       0000:0000:0000:0000:0000:ffff:0a00:021f  10.0.2.31    v2      ens15f0.2
mlx5_1          1       0       fe80:0000:0000:0000:ee0d:9aff:fe48:aeff               v1      ens15f1
mlx5_1          1       1       fe80:0000:0000:0000:ee0d:9aff:fe48:aeff               v2      ens15f1
mlx5_1          1       2       fe80:0000:0000:0000:141a:ad0d:abc6:8b65               v1      ens15f1
mlx5_1          1       3       fe80:0000:0000:0000:141a:ad0d:abc6:8b65               v2      ens15f1
mlx5_1          1       4       0000:0000:0000:0000:0000:ffff:0a00:0220  10.0.2.32    v1      ens15f1.2
mlx5_1          1       5       0000:0000:0000:0000:0000:ffff:0a00:0220  10.0.2.32    v2      ens15f1.2
n_gids_found=12
```

Index 5 of both devices are running RoCE v2 on VLAN2, which is what is needed.

> Check everything with tc_wrap.py. Make sure everything for VLAN 2 is priority 5.

```
#tc_wrap.py -i ens15f0.2
Traffic classes are set to 8
UP  0
UP  1
UP  2
UP  3
UP  4
UP  5
skprio: 0 (vlan 2)
skprio: 1 (vlan 2)
skprio: 2 (vlan 2 tos: 8)
skprio: 3 (vlan 2)
skprio: 4 (vlan 2 tos: 24)
skprio: 5 (vlan 2)
skprio: 6 (vlan 2 tos: 16)
skprio: 7 (vlan 2)
skprio: 8 (vlan 2)
skprio: 9 (vlan 2)
skprio: 10 (vlan 2)
skprio: 11 (vlan 2)
skprio: 12 (vlan 2)
skprio: 13 (vlan 2)
skprio: 14 (vlan 2)
skprio: 15 (vlan 2)
UP  6
UP  7
```

```
# tc_wrap.py -i ens15f01
Traffic classes are set to 8
UP  0
UP  1
UP  2
UP  3
UP  4
UP  5
skprio: 0 (vlan 2)
skprio: 1 (vlan 2)
skprio: 2 (vlan 2 tos: 8)
skprio: 3 (vlan 2)
skprio: 4 (vlan 2 tos: 24)
skprio: 5 (vlan 2)
skprio: 6 (vlan 2 tos: 16)
skprio: 7 (vlan 2)
skprio: 8 (vlan 2)
skprio: 9 (vlan 2)
skprio: 10 (vlan 2)
skprio: 11 (vlan 2)
skprio: 12 (vlan 2)
skprio: 13 (vlan 2)
skprio: 14 (vlan 2)
skprio: 15 (vlan 2)
UP  6
UP  7
```

> *Optional:* Configure ETS and DCQCN

With our newer OFED (We are using version 3.4.2.1.0.43101), ECN is enabled by default in the firmware. However, with an older OFED we would enter the following to enable it until next reboot:

```
#echo 1 > /sys/class/net/ens15f0/ecn/roce_np/enable/5
#echo 1 > /sys/class/net/ens15f0/ecn/roce_rp/enable/5
```

This would enable notification point and reaction point for priority 5 for all adapters.

For persistent configuration on older Mellanox OFED, the mlxconfig tool can be used from the Mellanox Firmware Tools package (MFT).

#tar zxvf mft-4.9.0-38-x86_64-rpm.tgz
#./mft-4.9.0-38-x86-64-rpm/install.sh
#mst start

Find the device name

#mst status

```
MST devices:
------------
/dev/mst/mt4119_pciconf0          - PCI
configuration cycles access.

domain:bus:dev.fn=0000:03:00.0 addr.reg=88
data.reg=92
                                   Chip revision
is: 00
```

#mlxconfig –d /dev/mst/mt4119_pciconf0 q | grep ROCE_CC_PRIO_MASK

```
ROCE_CC_PRIO_MASK_P1                 255
ROCE_CC_PRIO_MASK_P2                 255
```

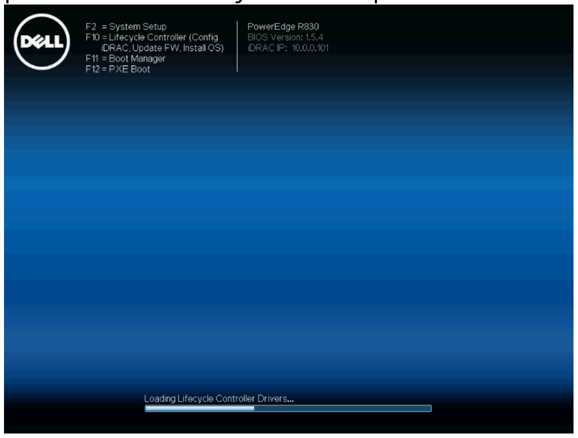
Each traffic class 7-0 has a bit in the mask, so we can see here that with our newer OFED ECN is enabled for all (11111111b=255). If we needed to permanently enable a particular traffic class in an older OFED, we would use hexadecimal. For example, enabling ECN for priority 5 would require the following:

#mlxconfig –d /dev/mst/mt4119_pciconf0 –y s ROCE_CC_PRIO_MASK_P1=0x20

Cavium

In order to use RoCE with Cavium cards, the DCBX and priority number are selected in the UEFI HII interface. Here is an example of this on a Dell server:

1. When the menu appears at boot in the top right, press F2 to enter System Setup.

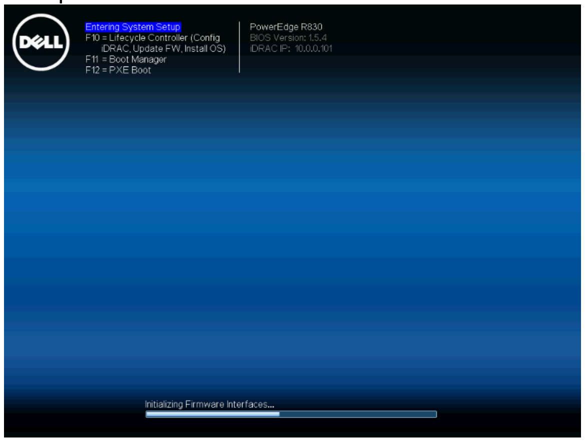


F2 = System Setup
F10 = Lifecycle Controller (Config iDRAC, Update FW, Install OS)
F11 = Boot Manager
F12 = PXE Boot

PowerEdge R830
BIOS Version: 1.5.4
iDRAC IP: 10.0.0.101

Loading Lifecycle Controller Drivers...

2. Wait while normal boot procedures are completed.

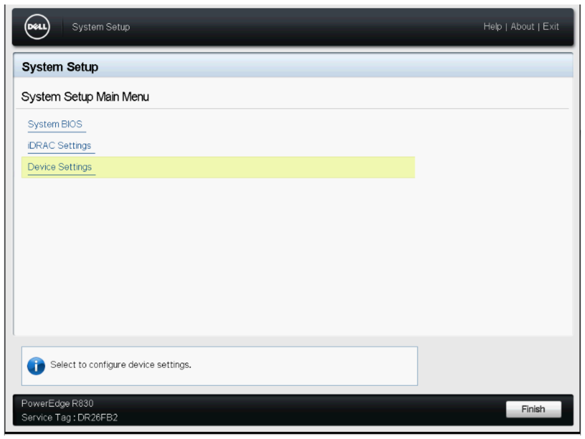


Entering System Setup
F10 = Lifecycle Controller (Config iDRAC, Update FW, Install OS)
F11 = Boot Manager
F12 = PXE Boot

PowerEdge R830
BIOS Version: 1.5.4
iDRAC IP: 10.0.0.101

Initializing Firmware Interfaces...

3. Select Device Settings at the System Setup Main Menu.



System Setup

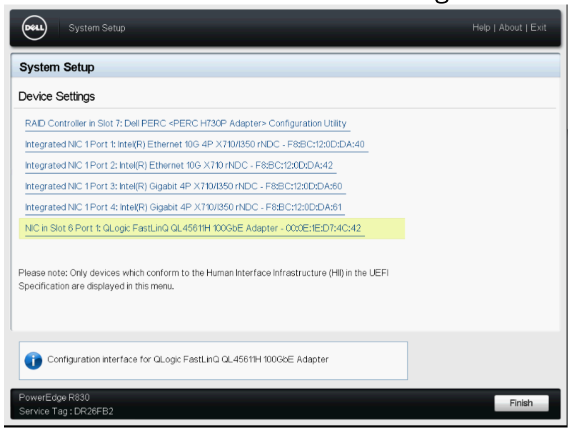
System Setup Main Menu

System BIOS
iDRAC Settings
Device Settings

Select to configure device settings.

PowerEdge R830
Service Tag : DR26FB2

4. Choose the Cavium Device to configure.



System Setup

Device Settings

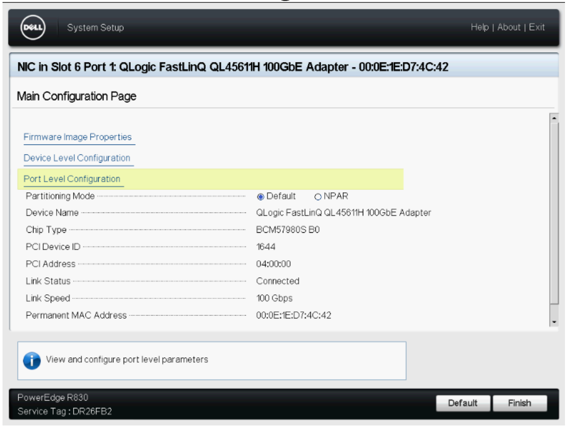
RAID Controller in Slot 7: Dell PERC <PERC H730P Adapter> Configuration Utility
Integrated NIC 1 Port 1: Intel(R) Ethernet 10G 4P X710/I350 rNDC - F8:BC:12:0D:DA:40
Integrated NIC 1 Port 2: Intel(R) Ethernet 10G X710 rNDC - F8:BC:12:0D:DA:42
Integrated NIC 1 Port 3: Intel(R) Gigabit 4P X710/I350 rNDC - F8:BC:12:0D:DA:60
Integrated NIC 1 Port 4: Intel(R) Gigabit 4P X710/I350 rNDC - F8:BC:12:0D:DA:61
NIC in Slot 6 Port 1: QLogic FastLinQ QL45611H 100GbE Adapter - 00:0E:1E:D7:4C:42

Please note: Only devices which conform to the Human Interface Infrastructure (HII) in the UEFI Specification are displayed in this menu.

Configuration interface for QLogic FastLinQ QL45611H 100GbE Adapter

PowerEdge R830
Service Tag : DR26FB2

5. Select Port Level Configuration.



NIC in Slot 6 Port 1: QLogic FastLinQ QL45611H 100GbE Adapter - 00:0E:1E:D7:4C:42

Main Configuration Page

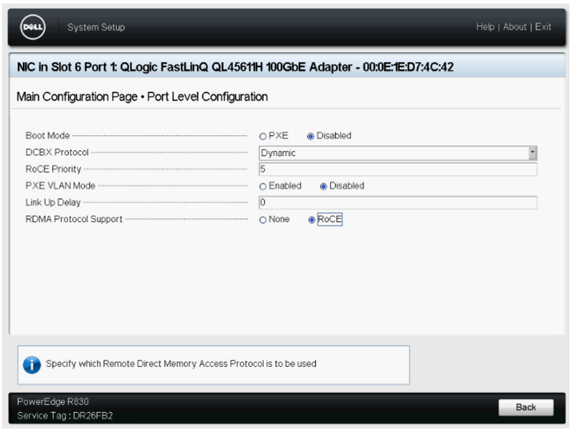
Firmware Image Properties
Device Level Configuration
Port Level Configuration

Partitioning Mode	[✓ Default] NPAR
Device Name	QLogic FastLinQ QL45611H 100GbE Adapter
Chip Type	BCM57980S B0
PCI Device ID	1644
PCI Address	04:00:00
Link Status	Connected
Link Speed	100 Gbps
Permanent MAC Address	00:0E:1E:D7:4C:42

View and configure port level parameters

PowerEdge R830
Service Tag : DR26FB2

9. Select Dynamic DCBX protocol, RoCE for RDMA Protocol support, and choose an appropriate RoCE Priority.
Optional: If DCQCN is going to be used, leave the RoCE priority set to 0 here.



NIC in Slot 6 Port 1: QLogic FastLinQ QL45611H 100GbE Adapter - 00:0E:1E:D7:4C:42

Main Configuration Page • Port Level Configuration

Boot Mode	PXE [✓ Disabled]
DCBX Protocol	Dynamic
RoCE Priority	5
PXE VLAN Mode	Enabled [✓ Disabled]
Link Up Delay	0
RDMA Protocol Support	None [✓ RoCE]

Specify which Remote Direct Memory Access Protocol is to be used

PowerEdge R830
Service Tag : DR26FB2

Other manufacturers should have similar UEFI hook-ins for the Cavium cards allowing the DCBX option and traffic priority class to be set. If any difficulty is encountered using the HII interface on a server, contact Cavium technical support for an engineering solution.

While in system setup, configure the following in the System BIOS for optimal performance later:

> Disable hyperthreading.

> Set the power management to performance and disable any Energy Performance Tuning or Energy Efficient Turbo settings.

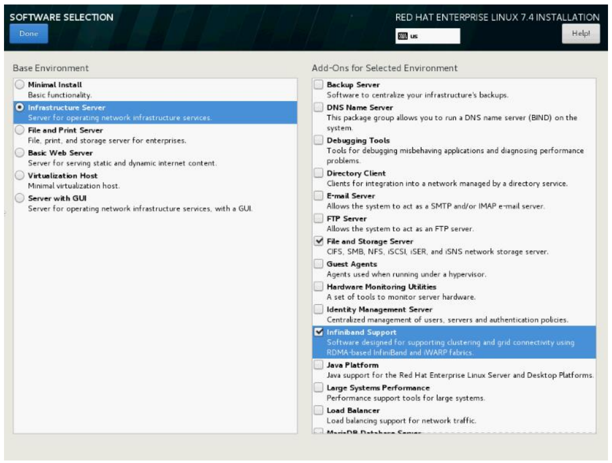
> Set Package C State Limit C0/C1 state.

> Disable CPU C3 report and CPU C6 report and Enhanced Halt State (C1E).

> Disable ACPI T-States.

Once the system has booted:

1. Install an InfiniBand OFED. Cavium recommends using the inbox OFED. This can be installed at OS installation by checking off "Infiniband Support."



Base Environment

- ○ **Minimal Install** Basic functionality.
- ● **Infrastructure Server** Server for operating network infrastructure services.
- ○ **File and Print Server** File, print, and storage server for enterprises.
- ○ **Basic Web Server** Server for serving static and dynamic internet content.
- ○ **Virtualization Host** Minimal virtualization host.
- ○ **Server with GUI** Server for operating network infrastructure services, with a GUI.

Add-Ons for Selected Environment

- ☐ **Backup Server** Software to centralize your infrastructure's backups.
- ☐ **DNS Name Server** This package group allows you to run a DNS name server (BIND) on the system.
- ☐ **Debugging Tools** Tools for debugging misbehaving applications and diagnosing performance problems.
- ☐ **Directory Client** Clients for integration into a network managed by a directory service.
- ☐ **E-mail Server** Allows the system to act as a SMTP and/or IMAP e-mail server.
- ☐ **FTP Server** Allows the system to act as an FTP server.
- ✓ **File and Storage Server** CIFS, SMB, NFS, iSCSI, iSER, and iSNS network storage server.
- ☐ **Guest Agents** Agents used when running under a hypervisor.
- ☐ **Hardware Monitoring Utilities** A set of tools to monitor server hardware.
- ☐ **Identity Management Server** Centralized management of users, servers and authentication policies.
- ✓ **Infiniband Support** Software designed for supporting clustering and grid connectivity using RDMA-based InfiniBand and iWARP fabrics.
- ☐ **Java Platform** Java support for the Red Hat Enterprise Linux Server and Desktop Platforms.
- ☐ **Large Systems Performance** Performance support tools for large systems.
- ☐ **Load Balancer** Load balancing support for network traffic.

If the OFED was not installed at OS installation, it can be downloaded and installed later.

```
# yum –y groupinstall "InfiniBand Support"
```

2. *Optional:* If using an older/compatible OS, install extra packages for configuration and testing. (These packages may conflict with the rdma_core package used in RHEL/CentOS 7):

```
#yum –y install perftest infiniband-diags
```

3. The RDMA drivers should have already been installed when the NIC drivers were installed during TCP/IP set-up and only require a modprobe (#modprobe qedr). However, if they were not installed or if an upgrade is required (find your current driver/firmware versions with #ethtool –i), download the updated drivers and firmware, and then:

   > Unload the old drivers

```
#lsmod | grep qed
#rmmod qedf (FCoE-Offload if present)
#rmmod qedi (iSCSI-Offload if present)
#rmmod qedr (RDMA-Offload if present)
#rmmod qede
#rmmod qed
#depmod –a
#rpm –e kmod-qlgc-fastlinq.x86_64
#unzip
```

   > Install the new drivers and firmware

```
Linux_RPM_QCScli_Package_45xxx_41xxx_8.33.9.
0-1.zip
#unzip Linux_FWupg_45xxx_2.10.44.zip
#rpm -ivh ./Linux_RPMs/rhel7u4/RPMS/kmod-
qlgc-fastlinq-8.33.9.0-1.rhel7u4.x86_64.rpm
```

#rpm –ivh ./Linux_RPMs/rhel7u4/RPMS/rdma-core-15-2.el7.x86_64.rpm
Loading (modprobe) the qedr module will load the dependent qede and qed modules
#modprobe qedr
Check to make sure all 3 modules are loaded. (#lsmod | grep qed)
#cd x86_64
#tar xvfz lnxfswnx2-x86_64.sdk.tgz
#chmod a+x LnxQlgcUpg.sh
Run the install shell script
# ./LnxQlgcUpg.sh
#shutdown –r now
Doublecheck upon reboot
#lsmod | grep qed
The qedr module will normally not load on boot up, so you should place "modprobe qedr" in your Linux startup script
#ethtool –i p6p1

4. Make sure the CPUs are set to performance mode. Check each processor.
   #cat /sys/devices/system/cpu/cpu*/cpufreq/scaling_governor

```
performance
performance
…
```

   It should read performance for all. If any processors are in powersave, manually change each of them to performance.
   # echo –n performance > /sys/devices/system/cpu/cpu0/cpufreq/scaling_governor
   …
5. Disable throttling.
   > in /etc/default/grub, find the end of the line "GRUB_CMDLINE_LINUX=" and add "intel_idle.max_cstate=0 processor.max_cstate=1 mce=ignore_ce idle=poll"

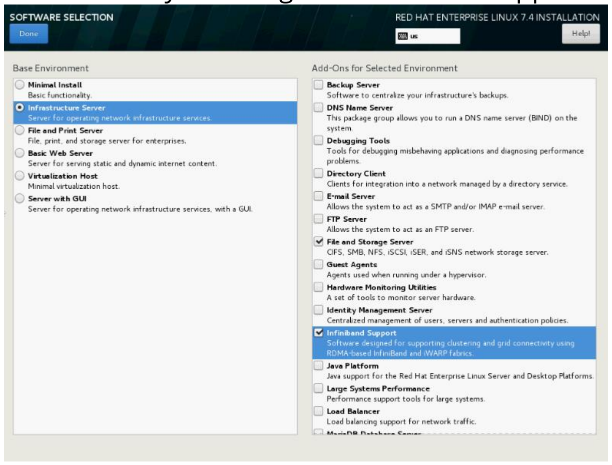
   > #grub2-mkconfig –o /boot/grub2/grub.cfg
6. Reboot. Do checks from step 4 again.
7. Disable and stop firewall.
   #systemctl disable firewalld

   #systemctl stop firewalld
8. *Optional*: Configure ECN
   The tools necessary for ECN configuration are contained in the driver source code download. Once obtained:
   #unzip Linux_Source_45xxx_41xxx_8.33.9.0.zip
   #tar xvf fastlinq-8.33.9.0.tar.bz2
   #cd fastlinq-8.33.9.0/add-ons/debug/debugfs
   #./debugfs.sh –n p6p1 –t dcbx_set_mode 0x10
   #./debugfs.sh –n p6p1 –t dcbx_set_pfc 5 1
   #./debugfs.sh –n p6p1 –t rdma_glob_ecn 1
   #./debugfs.sh –n p6p1 –t rdma_glob_vlan_pri 5
   #modprobe qed dcqcn_enable=1 dcqcn_notification_point=1 dcqcn_reaction_point=1
   #modprobe qedr
   Should the reader decide to use DSCP instead of vLAN, debugfs.sh would also be used to configure this. Refer to Cavium documentation for more information.

Broadcom

Please obtain the most recent RoCE setup recommendations from Broadcom. Below are the steps that we took while preparing this guide.

1. Install an InfiniBand OFED. Broadcom recommends using the inbox OFED. This can be installed at OS installation by checking off "Infiniband Support"



Base Environment
- Minimal Install: Basic functionality.
- [✓ Infrastructure Server]: Server for operating network infrastructure services.
- File and Print Server: File, print, and storage server for enterprises.
- Basic Web Server: Server for serving static and dynamic internet content.
- Virtualization Host: Minimal virtualization host.
- Server with GUI: Server for operating network infrastructure services, with a GUI.

Add-Ons for Selected Environment
- Backup Server: Software to centralize your infrastructure's backups.
- DNS Name Server: This package group allows you to run a DNS name server (BIND) on the system.
- Debugging Tools: Tools for debugging misbehaving applications and diagnosing performance problems.
- Directory Client: Clients for integration into a network managed by a directory service.
- E-mail Server: Allows the system to act as a SMTP and/or IMAP e-mail server.
- FTP Server: Allows the system to act as an FTP server.
- [✓ File and Storage Server]: CIFS, SMB, NFS, iSCSI, iSER, and iSNS network storage server.
- Guest Agents: Agents used when running under a hypervisor.
- Hardware Monitoring Utilities: A set of tools to monitor server hardware.
- Identity Management Server: Centralized management of users, servers and authentication policies.
- [✓ Infiniband Support]: Software designed for supporting clustering and grid connectivity using RDMA-based InfiniBand and iWARP fabrics.
- Java Platform: Java support for the Red Hat Enterprise Linux Server and Desktop Platforms.
- Large Systems Performance: Performance support tools for large systems.
- Load Balancer: Load balancing support for network traffic.
- MariaDB Database Server

   If the OFED was not installed at OS installation, it can be downloaded and installed later:
   # yum –y groupinstall "InfiniBand Support"

2. Install Prerequisite Packages for OFED and driver kernel module compilation:
   #yum install –y 'libibverbs*' libibverbs-devel qperf perftest infiniband-diags make gcc kernel kernel-devel autoconf aclocal libtool
   #systemctl enable rdma
   #reboot
3. Enable RDMA Modules upon reboot:
   Edit /etc/modules-load.d/modules to include the following lines:

```
ib_core
ib_cm
rdma_ucm
ib_ucm
ib_uverbs
ib_umad
rdma_cm
```

4. Install the NIC and RDMA Drivers:
   Acquire the latest Broadcom drivers and firmware and follow the documentation provided. For CentOS 7.4, we did the following:
   #unzip BCMCD_v20.8.4.1.zip
   #cd Linux/Linux_Driver
   #tar xf netxtreme-bnxt_en-1.8.29.tar.gz
   #cd netxtreme-bnxt_en-1.8.29/
   #make
   #make install
   #depmod –a
   #modprobe bnxt_en
   #modprobe bnxt_re

5. Install the RDMA Libraries
   #cd ../../Linux_RoCE/RoCE_Lib/
   #tar xf libbnxt_re-20.8.0.7.tar.gz
   #cd libbnxt_re-20.8.0.7/
   #sh autogen.sh
   #./configure
   #make
   #make install
   #sh –c "echo /usr/local/lib >> /etc/ld.so.conf"
   #ldconfig
   #ln –s /usr/local/etc/libibverbs.d/q /etc/libibverbs.d
   #cp bnxt_re.driver /etc/libibverbs.d

6. Install the firmware updating tool
   #cd ../../../bnxtnvm/Linux/
   #cp bnxtnvm /usr/local/bin
   #chmod +x /usr/local/bin/bnxtnvm
   Update initramfs:
   #dracut –f

7. If you were unable to before, configure your TCP/IP and VLAN connections and bring up your interfaces
8. Update your firmware:
   #bnxtnvm upgrade ../../NVRAM_Images/*.pkg –vvv –dev=ens1f1d1
   #bnxtnvm upgrade ../../NVRAM_Images/*.pkg –vvv –dev=ens1f0
   #reboot

9. Create your RoCE boot script
   Create /etc/ifup-roce-ens1f0.2 and /etc/ifup-roce-ens2f1.2

```
sleep 2
/sbin/modprobe bnxt_re
sleep 5

/sbin/ethtool -A $IFACE autoneg off rx off
tx off

mkdir -p /sys/kernel/config/rdma_cm/bnxt_re0
echo RoCE v2 >
/sys/kernel/config/rdma_cm/bnxt_re0/ports/1/
default_roce_mode

mkdir -p /sys/kernel/config/bnxt_re/bnxt_re0
cd
/sys/kernel/config/bnxt_re/bnxt_re0/ports/1/
cc/

echo -n 0x1 > tos_ecn #TOS for RoCE packets
echo -n 0x0 > vlan_tx_disable   #Set VLAN
mode
echo -n 0x0 > alt_vlan_pcp     #TOS for
notification packets

#Set parameters and enable
echo -n 0x1 > cc_mode
echo -n 0xc8 > tcp_cp
echo -n 0x3 > nph_per_state
echo -n 0x28 > rtt
echo -n 0x3 > g
echo -n 0x2 > init_cr
echo -n 0xa > init_tr
echo -n 0x1 > enable
echo -n 0x1 > apply
```

   #chmod 755 /etc/ifup-roce-ens1f0.2
   #chmod 755 /etc/ifup-roce-ens1f1.2
   Create /sbin/ifup-local

```
#!/bin/sh

export IFACE=$1
if [ -x /etc/ifup-roce-${IFACE} ];
then
  . /etc/ifup-roce-${IFACE}
fi
```

#chmod 755 /sbin/ifup-local

10. Restart your NICs
    #ifdown ens1f0.2
    #ifdown ens1f1.2
    #ifup ens1f0.2
    #ifup ens1f1.2

11. *Optional:* Configure ETS
    #yum install lldpad
    #systemctl enable lldpad
    #systemctl start lldpad
    #lldptool –L –i eno1f0 adminStatus=rxtx
    #lldptool –T –i ens1f0 –V APP app=5,1,35093
    #lldptool –T –i ens1f0 –V APP app=5,3,4791
    #lldptool –T –i ens1f0 –V ETS-CFG tsa="0:ets,1:ets,2:strict,3:strict,4:strict,5:strict,6:strict, 7:strict" up2tc=0:0,1:0,2:0,3:0,4:0,5:1,6:0,7:0 tcbw=10,90,0,0,0,0,0,0

Application Setup - iSER

When deploying iSER over RoCE, perform the following steps after switch configuration, OS install, and adapter configuration:

1. On the initiator system, install the iSCSI-initiator-utils (if it was not installed with the OS).
   #yum install iscsi-initiator-utils

2. Find out the initiator IQN for future reference.
   #cat /etc/iscsi/initiatorname.iscsi

```
InitiatorName=iqn.1994-
05.com.redhat:f88f40d62c69
```

3. Set up the target system.
   > install targetcli if it was not installed with the OS.
   #yum install targetcli

   > List the block devices and take note for future reference.
   #lsblk

   > Run targetcli. At the present time, the ls command should give us an empty tree as pictured below. As the target is configured, the tree will fill out. (If the tree is not empty, "\>clearconfig confirm=true") will clear it.
   #targetcli
   \>ls

```
targetcli shell version 2.1.fb41
Copyright 2011-2013 by Datera, Inc and
others.
For help on commands, type 'help'.
/> ls
o- /............................. [...]
  o- backstores.................. [...]
  | o- block ..... [Storage Objects: 0]
  | o- fileio..... [Storage Objects: 0]
  | o- pscsi  .... [Storage Objects: 0]
  | o- ramdisk  .. [Storage Objects: 0]
  o- iscsi  ............. [Targets: 0]
  o- loopback  .......... [Targets: 0]
  o- srpt  .............. [Targets: 0]
```

   > Create backstores out of the block devices. (In our case, we used NVMe drives.)
   \>cd backstores/block
   \>create name=<backstorename1>
     dev=/dev/<device name>
   \>create name=<backstorename2>
     dev=/dev/<device name>
   ... Repeat for additional backstores.

```
\>cd backstores/block
/backstores/block> create name=nvme0
dev=/dev/nvme0n1
 Created block storage object nvme0
using /dev/nvme0n1.
/backstores/block> create name=nvme1
dev=/dev/nvme1n1
 Created block storage object nvme1
using /dev/nvme1n1.
/backstores/block> create name=nvme2
dev=/dev/nvme2n1
 Created block storage object nvme2
using /dev/nvme2n1.
/backstores/block> create name=nvme3
dev=/dev/nvme3n1
 Created block storage object nvme3
using /dev/nvme3n1.
/backstores/block> create name=nvme4
dev=/dev/nvme4n1
 Created block storage object nvme4
using /dev/nvme4n1.
```

   > Create a target and give it LUNs created from the previously created backstores.

```
/backstores/block> cd /iscsi
/iscsi> create
/iscsi> cd iqn.2003-01.org.linux-
iscsi.dmrtk-srvr-
d.x8664:sn.cdf664e56d8c/tpg1/luns
/iscsi/iqn.20...d8c/tpg1/luns> create
/backstores/block/nvme0
Created LUN 0.
/iscsi/iqn.20...d8c/tpg1/luns> create
/backstores/block/nvme1
Created LUN 1.
/iscsi/iqn.20...d8c/tpg1/luns> create
/backstores/block/nvme2
Created LUN 2.
/iscsi/iqn.20...d8c/tpg1/luns> create
/backstores/block/nvme3
Created LUN 3.
/iscsi/iqn.20...d8c/tpg1/luns> create
/backstores/block/nvme4
Created LUN 4.
```

> Designate an IP address from the target server as a portal for the initiators to reach the target. Always designate port 3260 as this is the default port for iSCSI.
\>cd ../portals
\>delete 0.0.0.0 3260
\>create <target portal IP> 3260

```
/iscsi/iqn.20...d8c/tpg1/luns> cd
../portals
/iscsi/iqn.20.../tpg1/portals> delete
0.0.0.0 3260
  Deleted network portal 0.0.0.0:3260
/iscsi/iqn.20.../tpg1/portals> create
10.0.8.252 3260
  Using default IP port 3260
  Created network portal 10.0.8.252:3260.
```

> Enable iSER for each portal.
\>cd <target portal IP>:3260
\>enable_iser true

```
/iscsi/iqn.20.../tpg1/portals> cd
10.0.8.252:3260
/iscsi/iqn.20....0.8.252:3260 >
enable_iser true
  iSER enable now: True
```

> Designate which initiators can connect to this target, using the initiator name From Step 2.
\>cd ../../acls
\>create <initiator name>

```
/iscsi/iqn.20....0.8.254:3260> cd
../../acls
/iscsi/iqn.20...d8c/tpg1/acls> create
iqn.1994-05.com.Red Hat:2ffa3543d275
  Created Node ACL for iqn.1994-
05.com.Red Hat:2ffa3543d275
  Created mapped LUN 4.
  Created mapped LUN 3.
  Created mapped LUN 2.
  Created mapped LUN 1.
  Created mapped LUN 0.
```

> Check the configuration using ls to view the whole tree, then save and exit (as shown below).
\>ls
\>saveconfig
\>exit

```
/> ls
o- / ..................................................................[...]
  o- backstores .......................................................[...]
  | o- block .......................................[Storage Objects: 5]
  | | o- nvme0 ............................[/dev/nvme0n1 (1.5TiB) write-thru activated]
  | | o- nvme1 ............................[/dev/nvme1n1 (1.5TiB) write-thru activated]
  | | o- nvme2 ............................[/dev/nvme2n1 (1.5TiB) write-thru activated]
  | | o- nvme3.............................[/dev/nvme3n1 (1.5TiB) write-thru activated]
  | | o- nvme4.............................[/dev/nvme4n1 (1.5TiB) write-thru activated]
  | o- fileio ......................................[Storage Objects: 0]
  | o- pscsi .......................................[Storage Objects: 0]
  | o- ramdisk .....................................[Storage Objects: 0]
  o- iscsi ...................................................[Targets: 1]
  | o- iqn.2003-01.org.linux-iscsi.dmrtk-srvr-d.x8664:sn.cdf664e56d8c .........[TPGs: 1]
  |   o- tpg1 ...........................................[no-gen-acls, no-auth]
  |     o- acls ..................................................[ACLs: 1]
  |     | o- iqn.1994-05.com.redhat:f88f40d62c69......................[Mapped LUNs: 5]
  |     |   o- mapped_lun0 .....................................[lun0 block/nvme0 (rw)]
  |     |   o- mapped_lun1 .....................................[lun1 block/nvme1 (rw)]
  |     |   o- mapped_lun2 .....................................[lun2 block/nvme2 (rw)]
  |     |   o- mapped_lun3 .....................................[lun3 block/nvme3 (rw)]
  |     |   o- mapped_lun4 .....................................[lun4 block/nvme4 (rw)]
  |     o- luns ..................................................[LUNs: 5]
  |     | o- lun0 ..........................................[block/nvme0 (/dev/nvme0n1)]
  |     | o- lun1 ..........................................[block/nvme1 (/dev/nvme1n1)]
  |     | o- lun2 ..........................................[block/nvme2 (/dev/nvme2n1)]
  |     | o- lun3 ..........................................[block/nvme3 (/dev/nvme3n1)]
  |     | o- lun4 ..........................................[block/nvme4 (/dev/nvme4n1)]
  |     o- portals ..............................................[Portals: 1]
  |       o- 10.0.8.252:3260 ..............................................[iser]
  o- loopback ................................................[Targets: 0]
  o- srpt ....................................................[Targets: 0]
/> saveconfig
  Last 10 configs saved in /etc/target/backup.
  Configuration saved to /etc/target/saveconfig.json
/> exit
```

4. Start a connection from the initiator and login to the target. Once this is done, new devices should show up with lsblk.
   #iscsiadm –m discovery --op=new --op=delete --type sendtargets --portal <target portal IP> -I iser
   #iscsiadm –m node –l
   #lsblk

The above discovery command sometimes does not work the first time. Try discovering with iSCSI and then changing to iSER:

#iscsiadm –m discovery -–type sendtargets –p <target portal IP>
#iscsiadm –m node –T <target iqn obtained at discovery> -o update –n iface.transport_name –v iser

The new LUNs will be visible to the initiator.

For our testing, we ran vdbench against our LUNs raw, but in most cases, the user would want to employ parted and mkfs to implement file systems on the new devices.

iSER Performance Considerations:

> Deploy additional targets, 1 per LUN, for best performance, more control over initiator access and target portal interface to LUN associations.

> For multi-interface initiators, a separate iface can be made for each interface, with separate sessions for both:

#iscsiadm –m iface –I iser1 –o new

#iscsiadm –m iface –I iser2 –o new

#iscsiadm –m iface –I iser1 –o update –n iface.ipaddress –v <IP address for interface 2>

#iscsiadm –m iface –I iser2 –o update –n iface.apaddress –v <IPAddress for interface 2>

#iscsiadm –m iface –I iser1 –o update –n iface.transport_name –v iser

#iscsiadm –m iface –I iser2 –o update –n iface.transport_name –v iser

#iscsiadm –m discovery -–op=new -–op=delete -–type sendtargets -–portal <target IP> -I iser1

#iscsiadm -–mode node –targetname <target obtained from above command> --portal <target IP>,<number obtained from above command, usually 1> --login –I iser1

#iscsiadm –m discovery -–op=new -–op=delete -–type sendtargets –portal <target IP> -I iser2

#iscsiadm -–mode node –targetname <target obtained from above command> --portal <target IP>,<number obtained from above command, usually 1> --login –I iser2

List your iSER sessions:

#iscsiadm –m session

(If necessary, to logout of all iSER sessions to start over:

#iscsiadm –node –u)

Below is both a simple configuration example following the commands listed above, as well as an example of how to set up multiple iSCSI sessions for high performance:

```
SIMPLE CONFIGURATION EXAMPLE
# iscsiadm -m discovery --op=new --op=delete --type sendtargets --portal 10.0.8.253 -I iser
10.0.8.252:3260,1 iqn.2003-01.org.linux-iscsi.dmrtk-srvr-d.x8664:sn. cdf664e56d8c
#iscsiadm -m node -l
Logging in to [iface: iser, target: iqn.2003-01.org.linux-iscsi.dmrtk-srvr-d.x8664:sn.cdf664e56d8c, portal:
10.0.8.252,3260]
#lsblk
NAME           MAJ:MIN RM   SIZE RO TYPE MOUNTPOINT
sda              8:0    0   477G  0 disk
├─sda1           8:1    0   500M  0 part /boot
└─sda2           8:2    0 476.5G  0 part
  ├─rhel-root 253:0     0    50G  0 lvm  /
  ├─rhel-swap 253:1     0     4G  0 lvm  [SWAP]
  └─rhel-home 253:2     0 422.4G  0 lvm  /home
sdb              8:16   0   1.5T  0 disk
sdc              8:32   0   1.5T  0 disk
sdd              8:48   0   1.5T  0 disk
sde              8:64   0   1.5T  0 disk
sdf              8:80   0   1.5T  0 disk
sr0             11:0    1  1024M  0 rom
```

```
HIGH PERFORMANCE CONFIGURATION EXAMPLE
# iscsiaddm -m iface -I iser1 -o new
New interface iser1 added
# iscsiadm -m iface -I iser2 -o new
New interface iser2 added
# iscsiadm -m iface -I iser1 -o update -n iface.ipaddress -v 10.0.2.51
iser1 updated.
# iscsiadm -m iface -I iser2 -o update -n iface.ipaddress -v 10.0.2.52
iser2 updated.
# ifscsiadm -m iface -I iser1 -o update -n iface.transport_name -v iser
iser1 updated.
# iscsiadm -m iface -I iser2 -o update -n iface.transport_name -v iser
iser2 updated.
#iscsiadm -m discovery --op=new --op=delete --type sendtargets --portal 10.0.1.101 - I iser1
10.0.1.101:3260,1 iqn.2003-01.org.linux-iscsi.dmrtk-srvr-z.x8664:sn.t1
10.0.1.101:3260,1 iqn.2003-01.org.linux-iscsi.dmrtk-srvr-z.x8664:sn.t2
10.0.1.101:3260,1 iqn.2003-01.org.linux-iscsi.dmrtk-srvr-z.x8664:sn.t3
10.0.1.101:3260,1 iqn.2003-01.org.linux-iscsi.dmrtk-srvr-z.x8664:sn.t4
# iscsiadm --mode node --targetname iqn.2003-01.org.linux-iscsi.dmrtk-srvr-z.x8664:sn.t1 --portal
10.0.1.101,1 --login -I iser1
Logging in to [iface: iser1, target: iqn.2003-01.org.linux-iscsi.dmrtk-srvr-z.x8664:sn.t1, portal:
10.0.1.101,3260] (multiple)
Login to [iface: iser1, target: iqn.2003-01.org.linux-iscsi.dmrtk-srvr-z.x8664:sn.t1, portal:
10.0.1.101,3260] successful.
# iscsiadm --mode node --targetname iqn.2003-01.org.linux-iscsi.dmrtk-srvr-z.x8664:sn .t2 --portal
10.0.1.101,1 --login -I iser1
Logging in to [iface: iser1, target: iqn.1003-01.org.linux-iscsi.dmrtk-srvr-z.x8664:sn.t2, portal:
10.0.1.101,3260] (multiple)
Login to [iface: iser1, target: iqn.1003-01.org.linux-iscsi.dmrtk-srvr-z.x8664:sn.t2, portal:
10.0.1.101,3260] successful.
# iscsiadm -m discovery --op=new --op=delete --type sendtargets --portal 10.0.0.1 -I iser2
iscsiadm: This command will remove the record [iface: iser1, target: iqn.2003-01.org.linux-iscsi.dmrtk-srvr-
z.x8664:sn.d6a3be7341ab, portal: 10.0.1.101,3260], but a session is using it. Logout session then rerun
command to remove record.
iscsiadm: This command will remove the record [iface: iser1, target: iqn.1003-01.org.linux-iscsi.dmrtk-srvr-
z.x8664:sn.t2, portal: 10.0.1.101,3260], but a session is using it. Logout session then rerun command to
remove record.
10.0.1.101:3260,1 iqn.2003-01.org.linux-iscsi.dmrtk-srvr-z.x8664:sn.t1
10.0.1.101:3260,1 iqn.1003-01.org.linux-iscsi.dmrtk-srvr-z.x8664:sn.t2
10.0.1.101:3260,1 iqn.2003-01.org.linux-iscsi.dmrtk-srvr-z.x8664:sn.t3
10.0.1.101:3260,1 iqn.2003-01.org.linux-iscsi.dmrtk-srvr-z.x8664:sn.t4
 # iscsiadm --mode node --targetname iqn.2003-01.org.linux-iscsi.dmrtk-srvr-z.x8664:sn.t3 --portal
10.0.1.101,1 --login -I iser2
Logging in to [iface: iser2, target: iqn.2003-01.org.linux-iscsi.dmrtk-srvr-z.x8664:sn.t3, portal:
10.0.1.101,3260] (multiple)
Login to [iface: iser2, target: iqn.2003-01.org.linux-iscsi.dmrtk-srvr-z.x8664:sn.t3, portal:
10.0.1.101,3260] successful.
# iscsiadm --mode node --targetname iqn.2003-01.org.linux-iscsi.dmrtk-srvr-z.x8664:sn.t4 --portal
10.0.1.101,1 --login -I iser2
Logging in to [iface: iser2, target: iqn.2003-01.org.linux-iscsi.dmrtk-srvr-z.x8664:sn.t4, portal:
10.0.1.101,3260] (multiple)
Login to [iface: iser2, target: iqn.2003-01.org.linux-iscsi.dmrtk-srvr-z.x8664:sn.t4, portal:
10.0.1.101,3260] successful.
iscsiadm -m session
iser: [4] 10.0.1.101:3260,1 iqn.2003-01.org.linux-iscsi.dmrtk-srvr-z.x8664:sn.t1 (non-flash)
iser: [5] 10.0.1.101:3260,1 iqn.2003-01.org.linux-iscsi.dmrtk-srvr-z.x8664:sn.t2 (non-flash)
iser: [7] 10.0.1.101:3260,1 iqn.2003-01.org.linux-iscsi.dmrtk-srvr-z.x8664:sn.t3 (non-flash)
iser: [8] 10.0.1.101:3260,1 iqn.2003-01.org.linux-iscsi.dmrtk-srvr-z.x8664:sn.t4 (non-flash)
#lsblk
```

Application Setup - NFSoRDMA

When deploying NFSoRDMA via RoCE, perform the following steps after switch configuration, OS install, and adapter configuration:

On your Server:

1. Make sure your storage has the right attributes.
   #chmod 777 /mnt/myshare
2. Make sure NFSoRDMA is running
   #modprobe svcrdma
   #service nfs start
   #echo rdma 20049 > /proc/fs/nfsd/portlist

3. Set up your shares by adding them to /etc/exports (fsid may not be necessary in all configurations, but was for our ramdisk targets):

```
/mnt/myshare
10.0.2.31(rw,sync,fsid=1,insecure_locsk,no_
root_squash)
/mnt/myshare2
10.0.2.51(rw,sync,fsid=2,insecure_locks,no_
root_squash)
```

4. Export the shares
#exportfs –a

On Each Client

5. Load the rdma module for NFS client
#modprobe xprtrdma
6. Mount the NFS share
#mkdir /mnt/nfsmount1
#mount –o rdma,port=20049
10.0.1.101:/mnt/myshare /mnt/nfsmount1

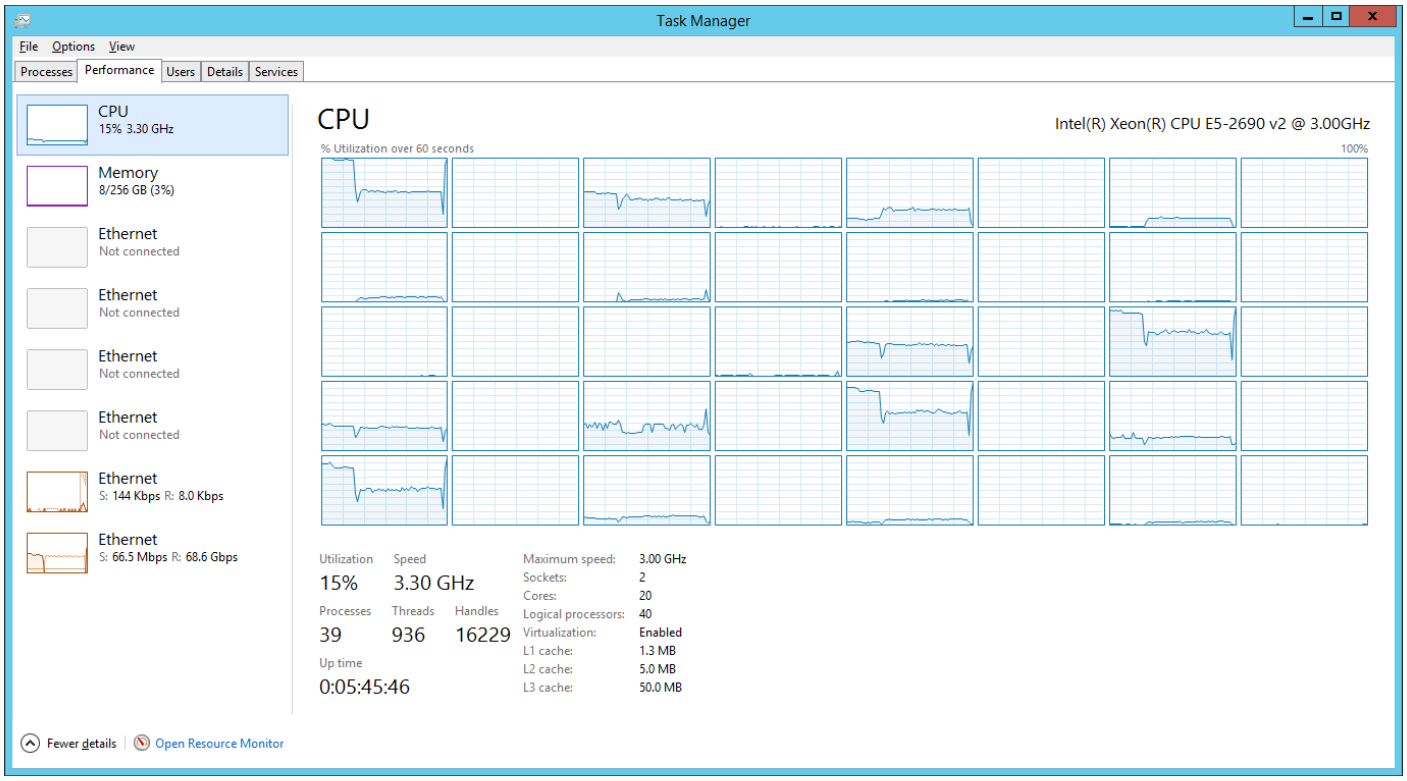
Test Results

Expected Performance from RoCE

When we replace TCP/IP with RoCE, we expect CPU consumption to decrease due to the Remote Direct Memory Access capabilities that eliminate CPU cycles used for data movement and transport layer processing. When using TCP/IP, despite configuring Receive Side Scaling (RSS) on NICs, certain CPU cores are often heavily utilized for transport processing while the rest of the cores are left idle.

Below is a typical example of what a server running SMB over TCP/IP might look like. Several cores are heavily utilized, while many are left idle. The server in our example is a larger, more powerful server.

Having an individual CPU core at maximum utilization can lead to some throughput degradation as the individual core cannot handle all of the traffic sent to it.

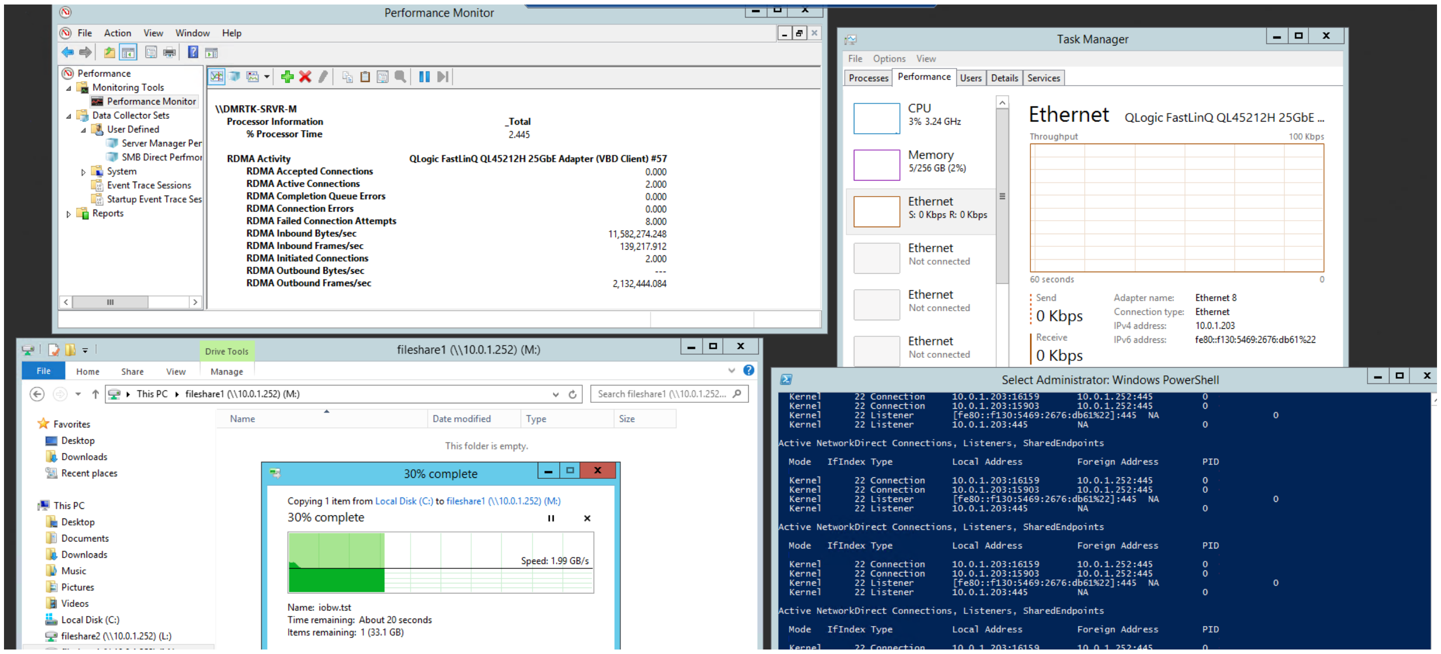


Utilization 15%
Speed 3.30 GHz
Processes 39
Threads 936
Handles 16229
Up time 0:05:45:46

Maximum speed:	3.00 GHz
Sockets:	2
Cores:	20
Logical processors:	40
Virtualization:	Enabled
L1 cache:	1.3 MB
L2 cache:	5.0 MB
L3 cache:	50.0 MB

A server with less processing available (either due to a small number of cores or due to other processes on the server consuming CPU cycles), can result in a throughput performance cap and increased latencies when TCP/IP is used. If RoCE is instead deployed on the same server, we would expect cores to be utilized more evenly and less CPU to be required for the same or greater amounts of traffic. If RoCE is deployed on a large server or a server with few other processes, we may notice the same throughput performance increase due to better traffic distribution between the cores. In most cases, this leads to a greater amount of throughput, and in some cases, the throughput is so much greater that the total CPU consumption will increase.

Below is a typical example of what a server running SMB Direct over RoCE might look like. Notice that the Ethernet is not registering any traffic, however a file is being copied across the network and RDMA traffic is showing up in Perfmon. To the right is netstat, showing an active SMB Direct Connection.


Performance Monitor
Task Manager
Ethernet QLogic FastLinQ QL45212H 25GbE ...
CPU 3% 3.24 GHz
Memory 5/256 GB (2%)
Ethernet S: 0 Kbps R: 0 Kbps
Ethernet Not connected
Send 0 Kbps
Receive 0 Kbps
fileshare1 (\\10.0.1.252) (M:)
30% complete
Speed: 1.99 GB/s
Select Administrator: Windows PowerShell

```
\\DMRTK-SRVR-M
  Processor Information                                  _Total
    % Processor Time                                      2.445

  RDMA Activity          QLogic FastLinQ QL45212H 25GbE Adapter (VBD Client) #57
    RDMA Accepted Connections                                          0.000
    RDMA Active Connections                                            2.000
    RDMA Completion Queue Errors                                       0.000
    RDMA Connection Errors                                             0.000
    RDMA Failed Connection Attempts                                    8.000
    RDMA Inbound Bytes/sec                                    11,582,274.248
    RDMA Inbound Frames/sec                                      139,217.912
    RDMA Initiated Connections                                         2.000
    RDMA Outbound Bytes/sec                                              ---
    RDMA Outbound Frames/sec                                   2,132,444.084
```

```
Adapter name:      Ethernet 8
Connection type:   Ethernet
IPv4 address:      10.0.1.203
IPv6 address:      fe80::f130:5469:2676:db61%22
```

```
Copying 1 item from Local Disk (C:) to fileshare1 (\\10.0.1.252) (M:)
30% complete
Name: iobw.tst
Time remaining: About 20 seconds
Items remaining: 1 (33.1 GB)
```

```
  Kernel      22 Connection     10.0.1.203:16159       10.0.1.252:445         0
  Kernel      22 Connection     10.0.1.203:15903       10.0.1.252:445         0
  Kernel      22 Listener       [fe80::f130:5469:2676:db61%22]:445  NA                    0
  Kernel      22 Listener       10.0.1.203:445         NA                     0

Active NetworkDirect Connections, Listeners, SharedEndpoints

  Mode   IfIndex Type           Local Address          Foreign Address        PID

  Kernel      22 Connection     10.0.1.203:16159       10.0.1.252:445         0
  Kernel      22 Connection     10.0.1.203:15903       10.0.1.252:445         0
  Kernel      22 Listener       [fe80::f130:5469:2676:db61%22]:445  NA                    0
  Kernel      22 Listener       10.0.1.203:445         NA                     0

Active NetworkDirect Connections, Listeners, SharedEndpoints

  Mode   IfIndex Type           Local Address          Foreign Address        PID

  Kernel      22 Connection     10.0.1.203:16159       10.0.1.252:445         0
  Kernel      22 Connection     10.0.1.203:15903       10.0.1.252:445         0
  Kernel      22 Listener       [fe80::f130:5469:2676:db61%22]:445  NA                    0
  Kernel      22 Listener       10.0.1.203:445         NA                     0

Active NetworkDirect Connections, Listeners, SharedEndpoints

  Mode   IfIndex Type           Local Address          Foreign Address        PID

  Kernel      22 Connection     10.0.1.203:16159       10.0.1.252:445         0
```

A great way to look at this interplay of CPU utilization and network throughput is through the CPU effectiveness: a ratio of throughput to percent of processor used. CPU effectiveness data can be used to extrapolate how much processor can be utilized on the server for other processes before it causes a network throughput cap due to a shortage of CPU resources.

In our testing, our throughput tests were the only processes other than general OS processes running on the server, so we may see how much throughput 1% of server CPU resources are capable of generating with each protocol. We would expect that, due to the offloading of network processing, CPU effectiveness will be higher for the RoCE protocol.

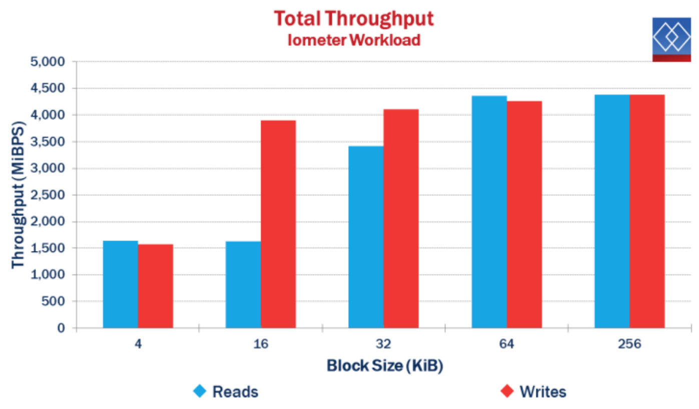
Expected Performance from RoCEv2 Congestion Management and ETS

As ETS grants preferred traffic a guaranteed bandwidth, we would expect that when it is added to a congested environment that throughput for the preferred traffic type – RoCE in our case – would increase.

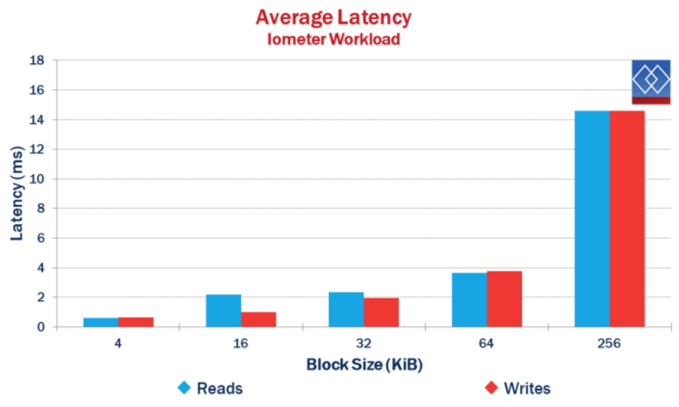
The addition of RoCEv2 Congestion Management to a congested environment should eliminate the need for many pauses on our lossless RoCE traffic and instead allow the sender to decrease the data rate. In a multihop configuration, not all intermediate hops will have to be paused at once, meaning that one host may be able to continue to send while another host adjusts the data rate instead of both hosts pausing. This should lead to a net increase of throughput in our lossless traffic.

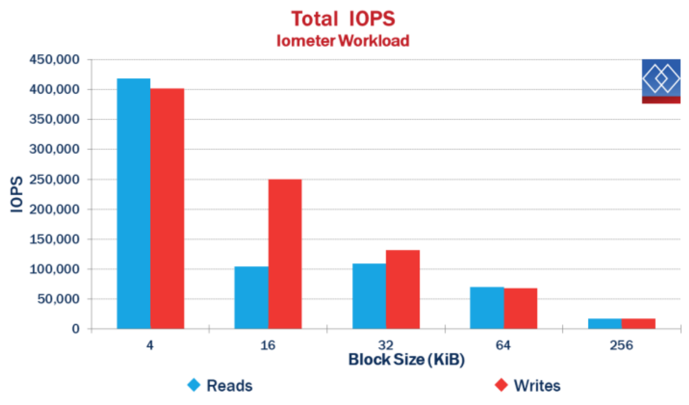
Windows Storage Spaces with Iometer

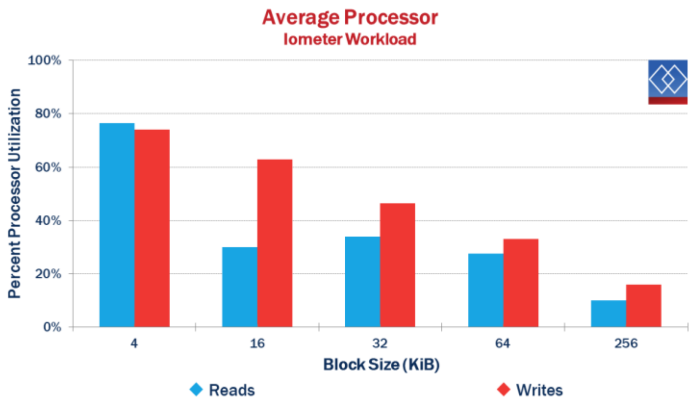
File shares were set up on ramdisks on one server connected to the Mellanox switch. The Mellanox switch had a 40GbE connection to the HPE switch and then the client server was connected to the HPE switch. Read and write traffic was generated with Iometer. It is easy to see here that the 40Gbps connection between the two servers was easily maxed out using the RoCE protocol.



Traffic gets close to 40Gbps line rate at a block size of 32KiB. Larger block sizes show an increase in latency as the 40Gbps link experiences congestion and sends pauses.







Processor effectiveness varies with block size. Here we see low processor effectiveness for 4KiB block sizes due to high IOPS producing high processor utilization. Processor utilization decreases and throughput increases with larger block sizes, producing a processor efficiency above 400 MBPS/%processor. If the 40Gbps bottleneck were removed, we would see the throughput increase again and the processor efficiency get even higher.

Windows Storage Spaces Direct with VMFleet – Mixed Vendor Configurations

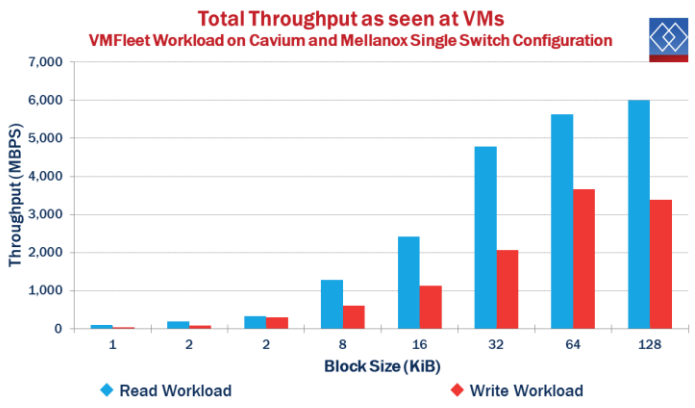
All of our Mixed-Vendor VMFleet configurations have two servers in a mirrored S2D cluster, both connected to one Mellanox switch. In such a configuration each cluster node has a complete local copy of all VM data. Therefore for VM read requests, a local copy of the data can usually be obtained faster than going over the network and therefore little RoCE traffic will be seen. However, for VM write requests, the write will have to be made to each node, updating both the local and the redundant copy of the data. For all write throughput seen by the VMFleet, we expect corresponding RoCE traffic to be seen on the network.

Seeing as our available NVMe throughput for eight drives is very high, we expect the total read capability of our VMFleet to be limited by the processor capabilities of the host instead of the storage performance. However, we expect the total write capability of each server to be limited by the NVMe storage for two reasons. First, NVMe storage tends to have write throughput that is half of read throughput. Second, each node is writing the data from its own VMs in addition to writing the data received from the other cluster node to update the redundant copies of the VM data for the other node.

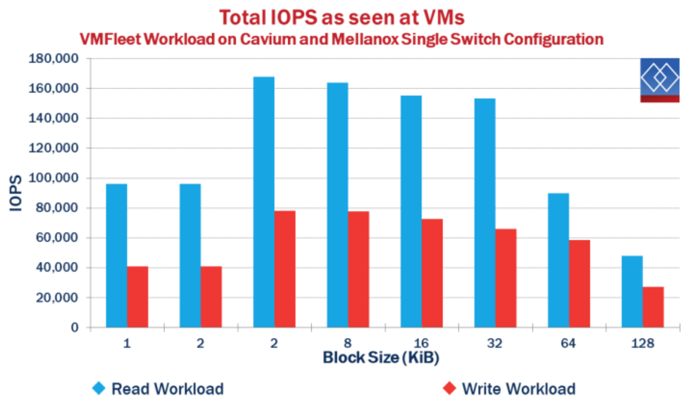
As reads will not generate much network throughput and writes will be storage-limited, we do not expect the VMFleet to be capable of achieving network line rate in this configuration.

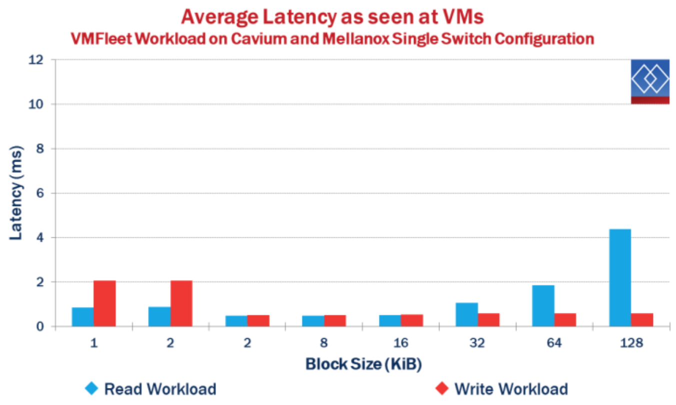
It is important to keep in mind that much of the data presented is as seen from the VMs. When the fleet writes, data for redundant copies will be sent by each node, producing bi-directional traffic. Line rate for writes, from the VMFleet perspective, will appear to be twice that of the network line rate due to the bi-directionality.

As well as providing a baseline for a non-network-bottlenecked VMFleet with RoCE, the Mixed-Vendor tests are designed to show interoperability of different manufacturer adapters and switches when using RoCE. The Mellanox-Cavium configuration has a 100GbE Mellanox adapter in one cluster node and a Cavium 4x25GbE adapter in the other. All 4 Cavium ports are teamed together in a SET switch, so this configuration has equal bandwidth available to both nodes.

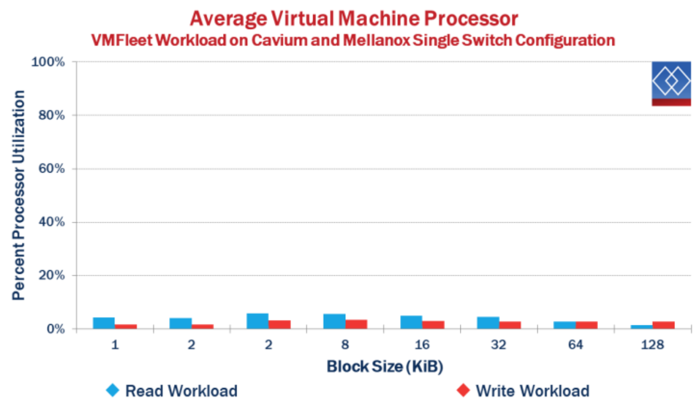


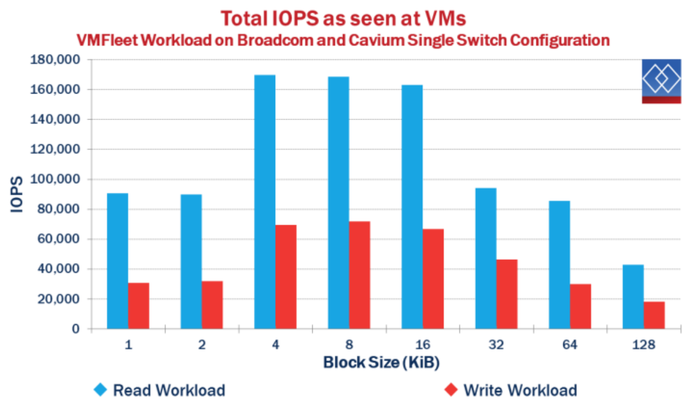
Additional tuning for later configurations will show an increase in total bandwidth, however starting at 6000MBPS reads is good.



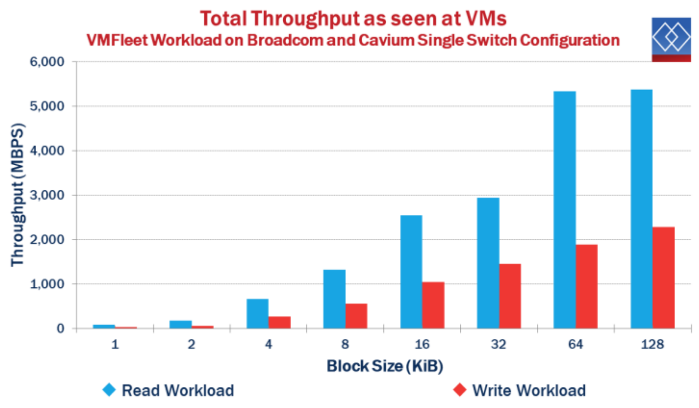


Latency, again, is as observed from the VMs, and will include all hypervisor overhead as well as some margin of error due to the clock skew inherent in virtualized guest machines.



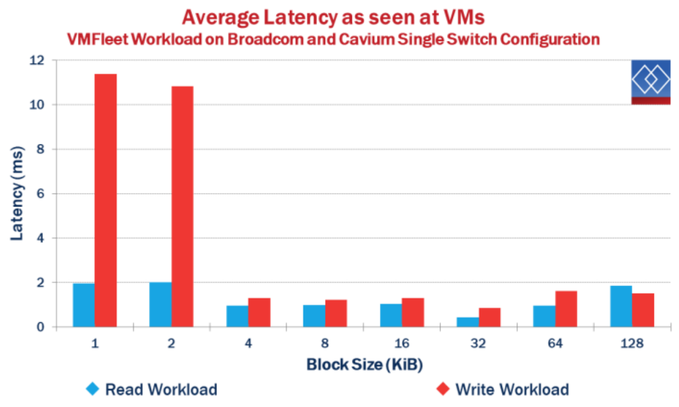


The Cavium-Broadcom configuration has a 100GbE Cavium adapter in one cluster node and a 2x25GbE Broadcom adapter in the other. As before, this shows the interoperability between different vendors. Furthermore, this configuration shows the resiliency when one cluster node has half the bandwidth of the other. Pause frames will constantly be sent from the Broadcom cluster node to the Cavium cluster node but the VMFleet does not fall down.



Here we see a slightly lower total bandwidth for the same VMFleet test, most likely due to pauses.

Total IOPS does not seem to suffer.



We see some increased latency for small block size write. It may be possible some coalescing is going on. We did see this pattern, less pronounced, with the equal bandwidth configuration as well.

Windows Storage Spaces Direct with VMFleet – The Effects of ETS and RoCEv2 Congestion Management on Congestion

All Congestion testing VMFleet configurations have two servers in the same mirrored S2D cluster as the Mixed-Vendor VMFleet configurations. The same considerations apply when viewing data from the VMs, and the previous section on Mixed-Vendor VMFleet testing should be reviewed before viewing the results of this section.

The following changes were made to the configuration for congestion testing:

> Originally for Mixed-Vendor VMFleet, each VM was assigned 1vCPU, in the hopes that there

were enough VMs to fully utilize all available host CPU. However it was noticed that hypervisor processor load was not evenly distributed and VMs were not fully utilizing the available processor. The configuration was changed to allow each VM multiple vCPU while limiting the total host resources available to each VM. This tuning increased processor utilization and nearly doubled throughput and IOPS.

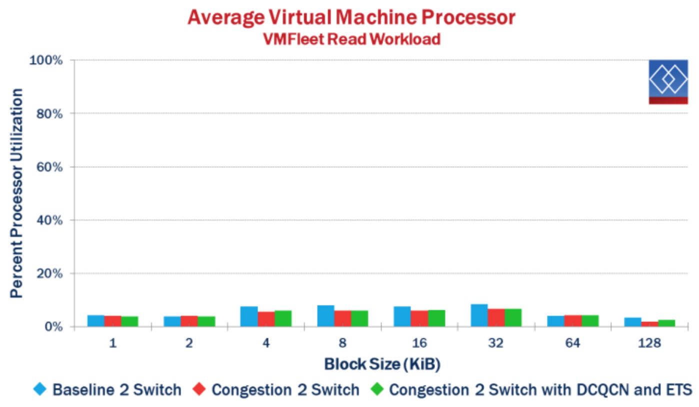
> Two pairs of congestion servers (four total), were added, with two servers attached to each switch. Each server sent TCP/IP congestion traffic to its partner on the other switch using iperf.

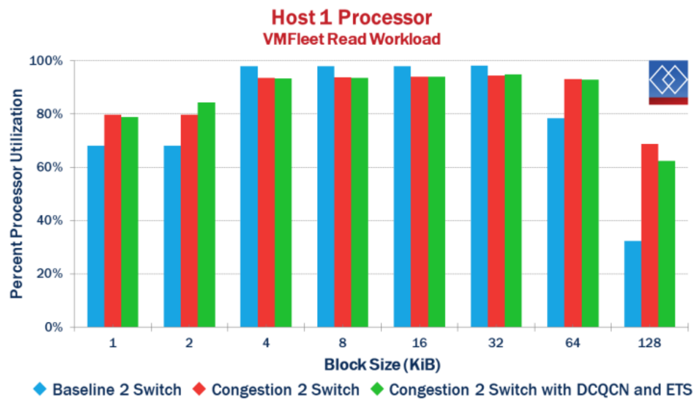
> Originally both cluster nodes were attached to the same switch. In the congestion tests, one cluster node was attached to the Mellanox switch and the other attached to the HPE switch. The two switches had a 40GbE link between them. The 40GbE link was meant to produce a bottleneck, and the 2-switch hop between the cluster nodes should cause problems. The nature of PFC is that each intermediary hop has to be paused in turn before the sender receives indication of congestion. This 2-switch configuration, with additional TCP/IP congestion added, should show performance degradation under PFC. With ETS/RoCEv2 Congestion Management added to help with congestion, some performance loss should be recovered.

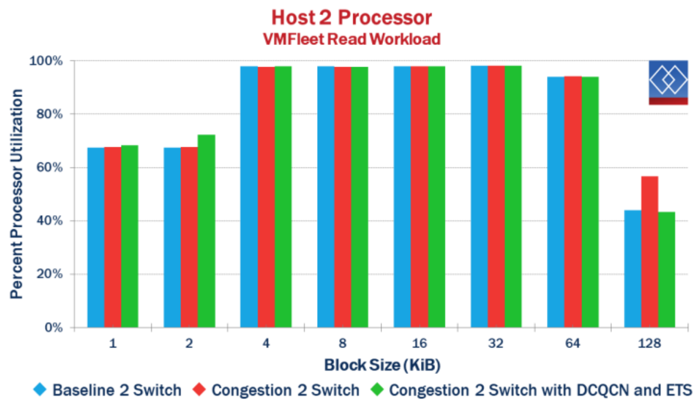
For our congestion tests, 3 sets of tests were run.

1. **Baseline Configuration:** First the adapters and switches were configured for VLAN and PFC, with losslessness for RoCE. The VMFleet tests were run with no congestion present on the network.
2. **Congestion 2 Switch:** The baseline test was repeated, only with the 4 congestion servers sending TCP/IP traffic during the VMFleet tests.
3. **Congestion 2 Switch with DCQCN and ETS:** Adapters and switches for the S2D Cluster were configured to use DCQCN and ETS. The 4 congestion servers once again sent TCP/IP traffic during the VMFleet tests.

To delve further into the effects of congestion and DCQCN, hypervisor data was obtained, as well as the sum of total traffic observed in each direction by all servers.

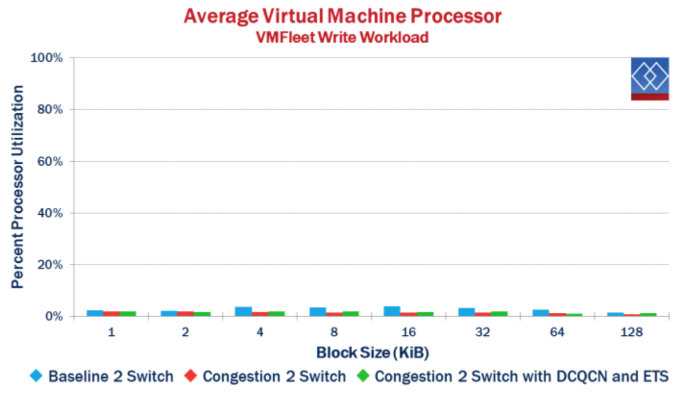


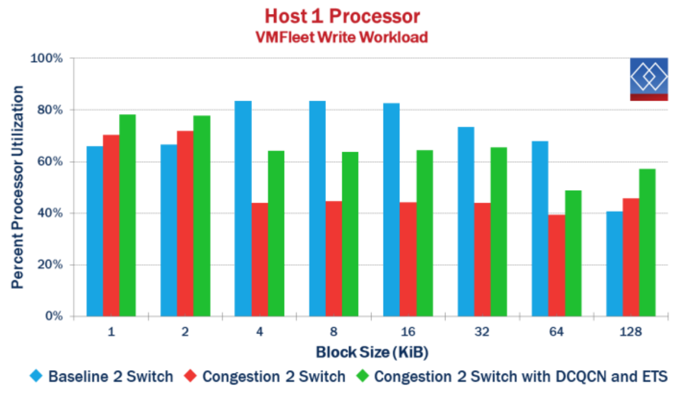


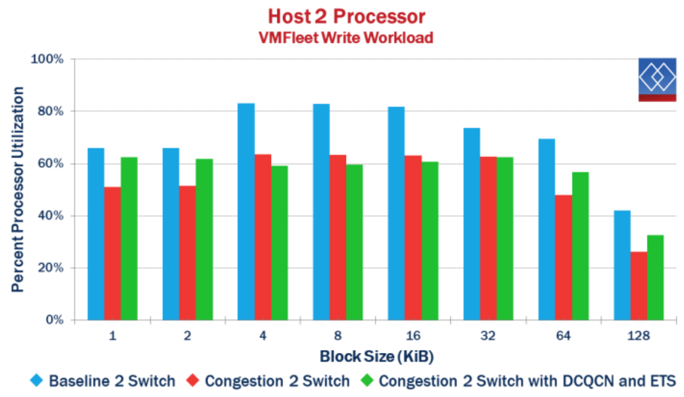


We can see that while the VMs only saw themselves using a small amount of the processor available to them, that the hypervisor processor, overall, was very high, especially for reads.

We also notice that with reads total processor utilization tends to drop slightly once congestion is introduced, but never recovers. It looks as though one host was affected by the congestion more than the other. We will see later on that one direction had more congestion than the other. Most likely only the direction with the higher amount of congestion traffic had a significant impact on our read workload.

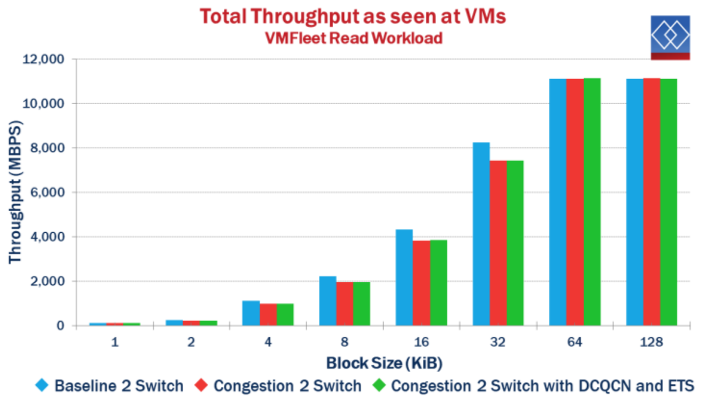


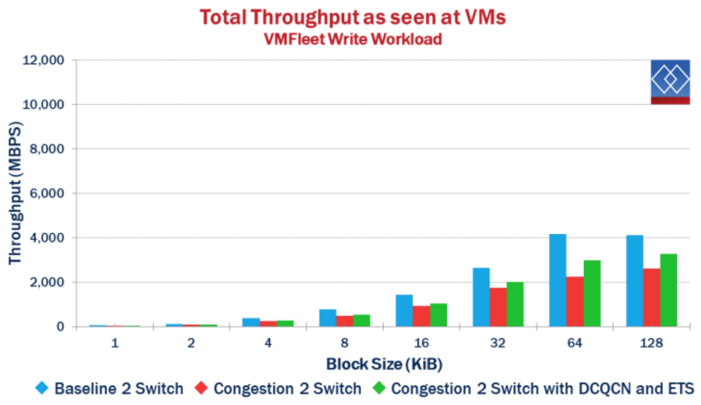




We see less processor utilization, from the VM perspective as well as the hypervisor, for writes. This is most likely because our write configurations have a storage bottleneck while our read configurations have a processor bottleneck.

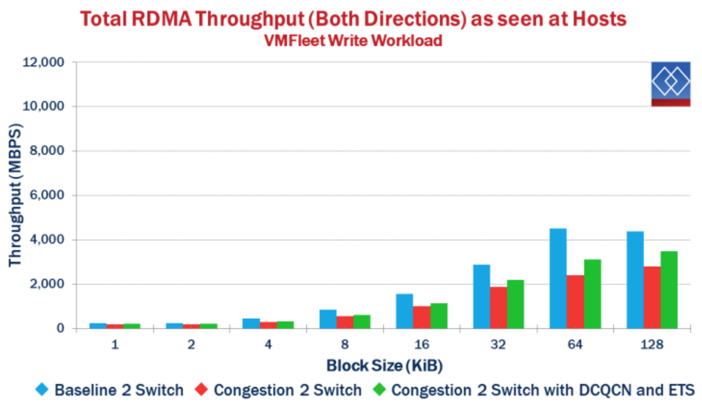
Similar to reads, with writes, we also notice that one host appears more affected than the other by the congestion. This host appears to recover some processor utilization when DCQCN and ETS is introduced, although this pattern is seen to some extent in both servers. We will see later that these processor changes tightly correspond to changes in throughput.



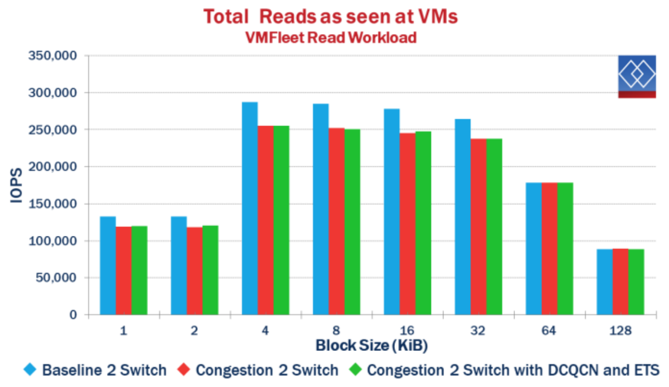


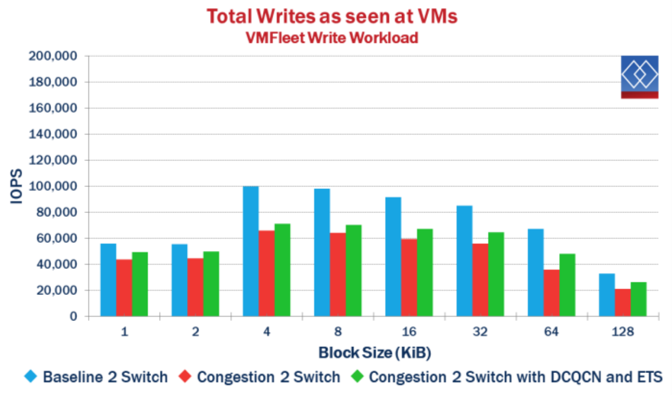
Total throughput, as seen by the VMFleet, decreases when congestion is introduced. Some throughput loss is recovered when DCQCN and ETS are introduced.

If we look at the total sum of throughput in both directions as seen by the cluster nodes instead of the VMs, we will see a simliar, but slightly higher, throughput. The host throughput is probably higher due to overhead. The percentage of host traffic due to overhead appears to be dependent opon block size.

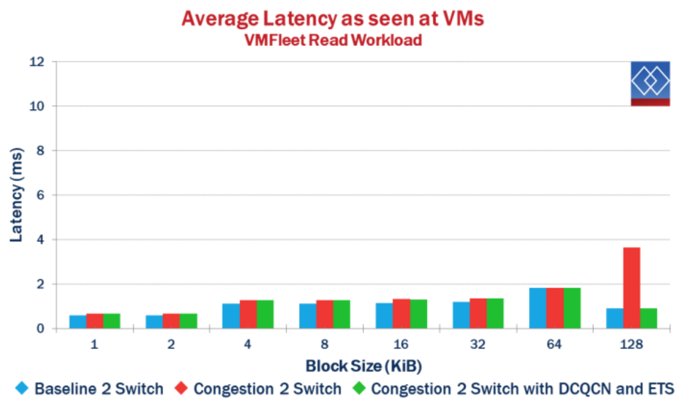


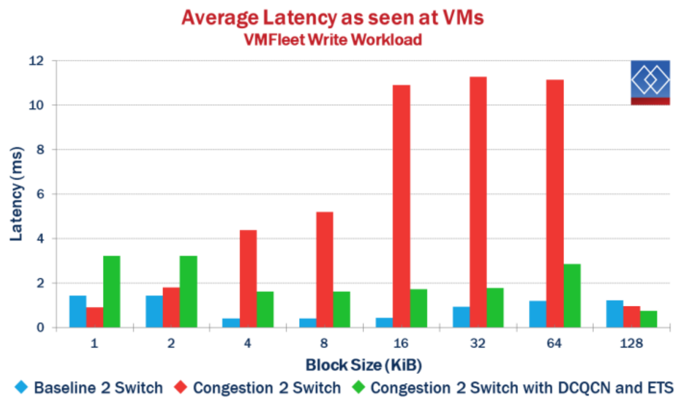
IOPS will follow a similar pattern as throughput.





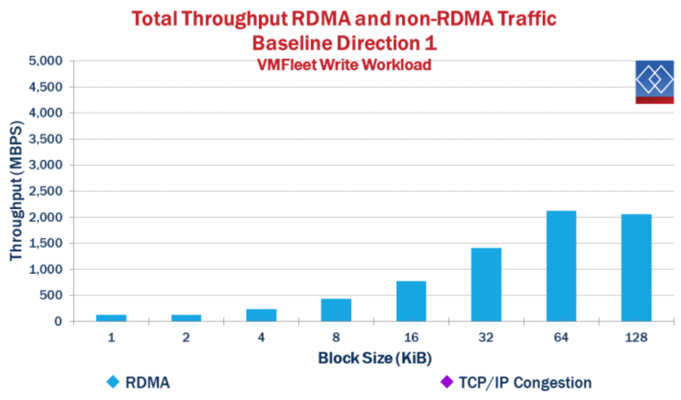
IOPS and throughput have doubled since we tuned our vCPU configuration

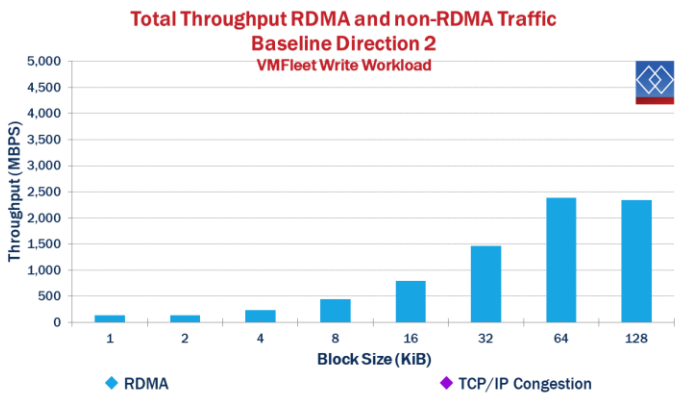


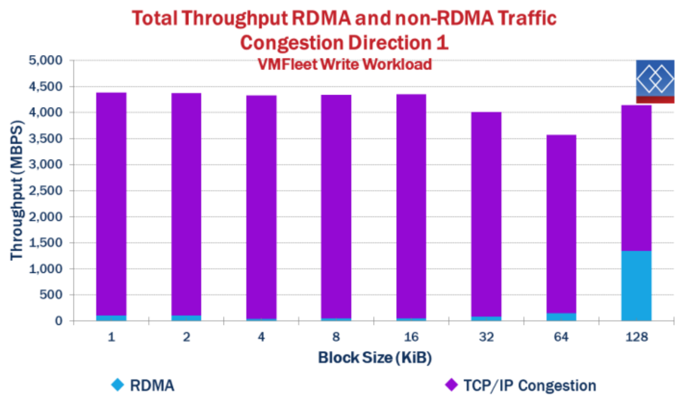


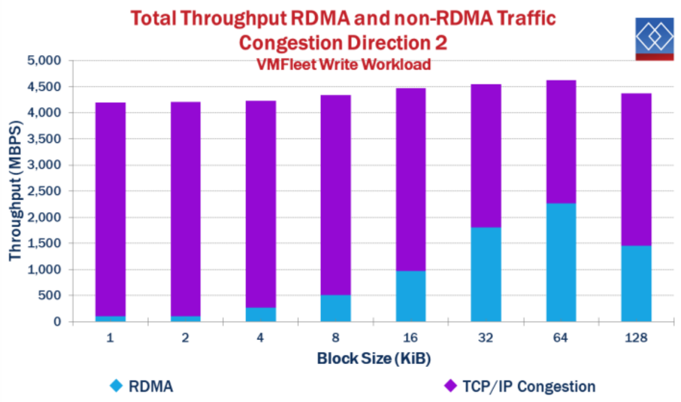
Again, the read workload did not produce much network bandwidth, so latency only increased slightly when congestion was added. However, for most block sizes, the write workload saw a large jump in latency when congestion was introduced. We also see a large reduction in latency when DCQCN and ETS are introduced to help lessen the effects of congestion.

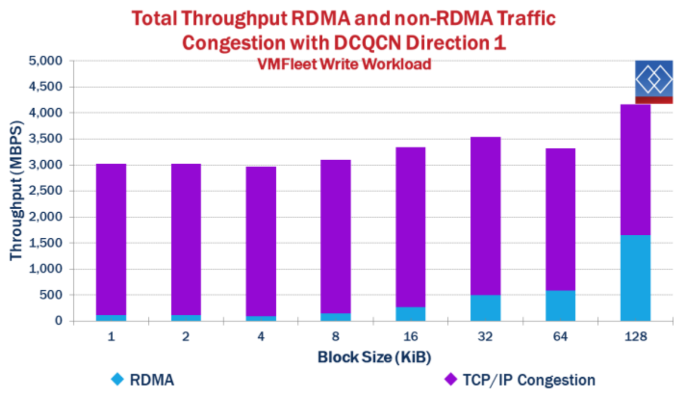
A breakdown of bandwidth by traffic type is useful for the write workload. We can see that the introduction of congestion severely impacts the RoCE in direction 1, most likely due to the higher volume of TCP/IP bandwidth in direction 1. This also pushes more RoCE in direction 2. The introduction of ETS and DCQCN limits the TCP/IP traffic in direction 1 and increases the RoCE traffic in both directions.

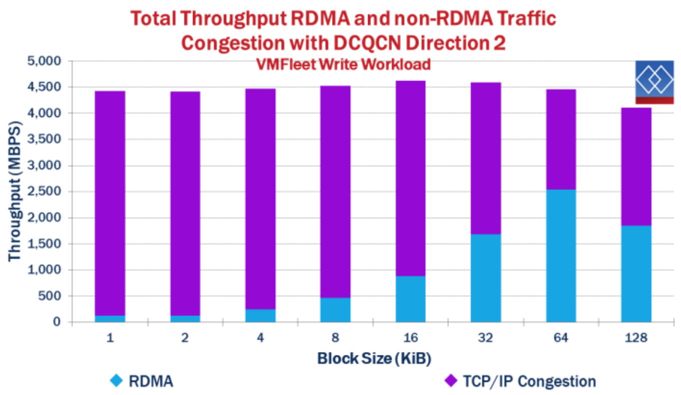










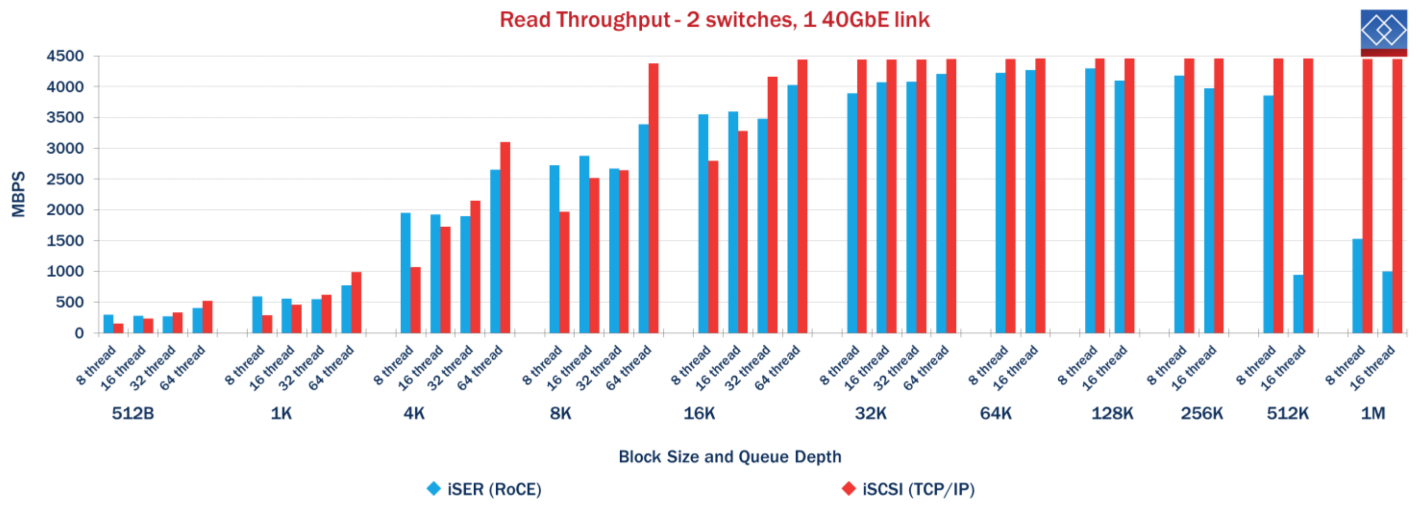


Mixed-Vendor RoCEv2 Congestion Management: Linux iSCSI with vdbench

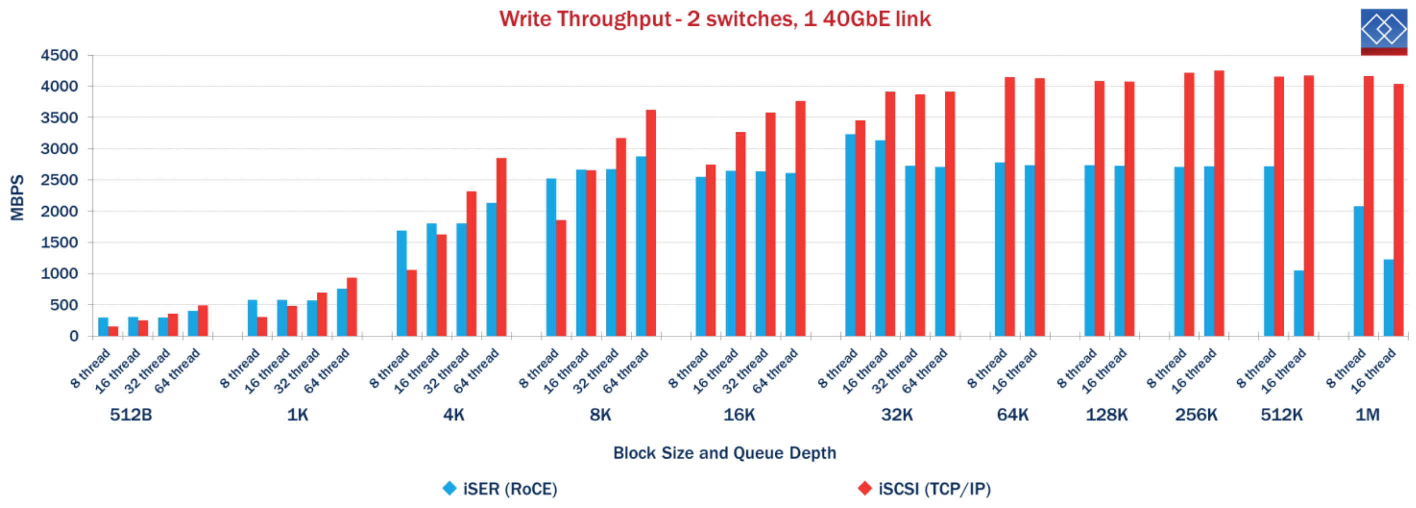
For our Linux tests, we deployed three servers. Our storage provided ramdisk targets and had a Cavium adapter, while one of our initiators/clients had a Mellanox adapter and the other had a Broadcom adapter. Our clients were connected to the Mellanox switch while our storage target was connected to the HPE switch. The switches were again connected with 40GbE to produce a bottleneck and opportunity to show off RoCEv2 Congestion Management. Both clients connected to the storage target to perform I/O with vdbench.

Each adapter and switch was configured for VLAN PFC, ETS, and DCQCN, showing vendor interoperability with DCQCN.

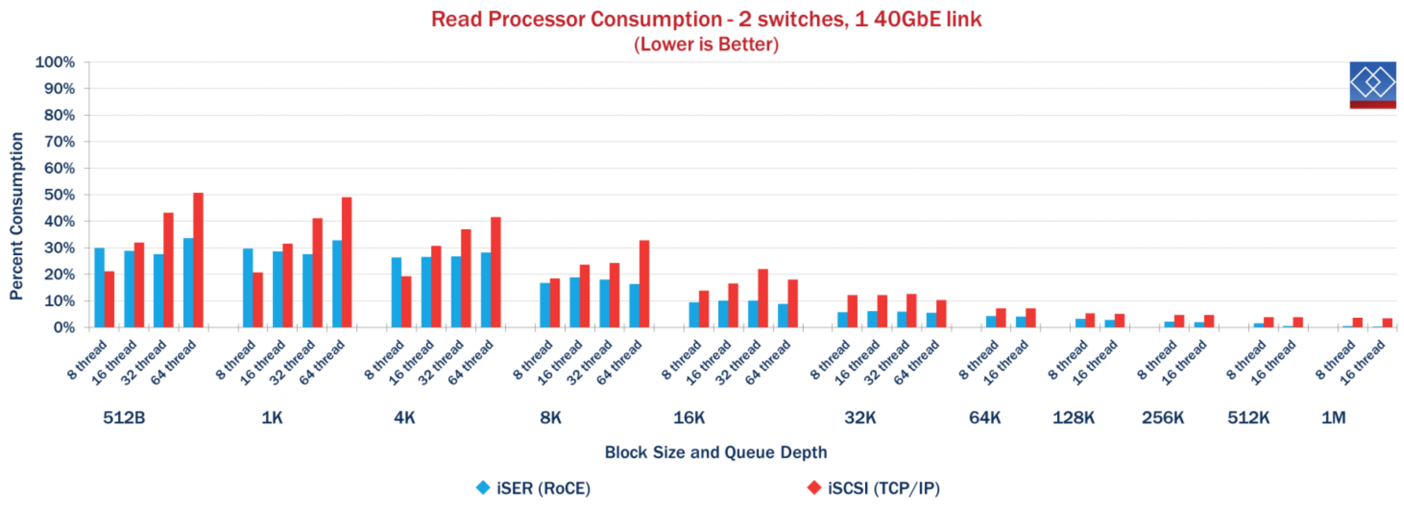
A high-performance setup was used, where each client port had two iSCSI sessions with the target, for a total of eight sessions.

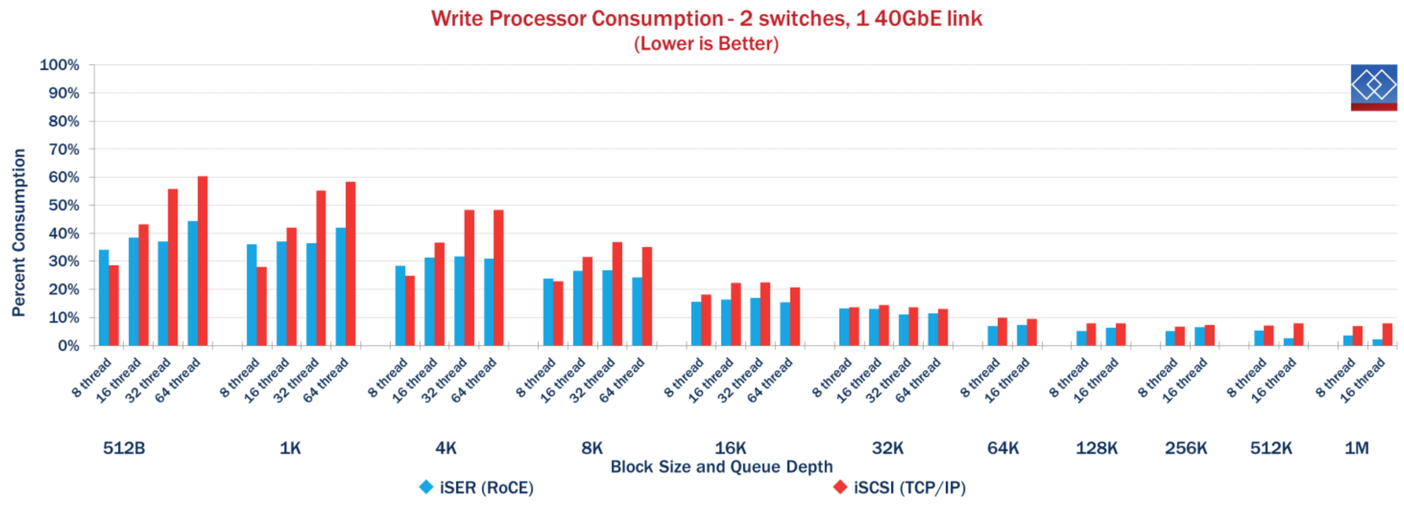


iSER and iSCSI both appear to easily reach line rate with our read workload.

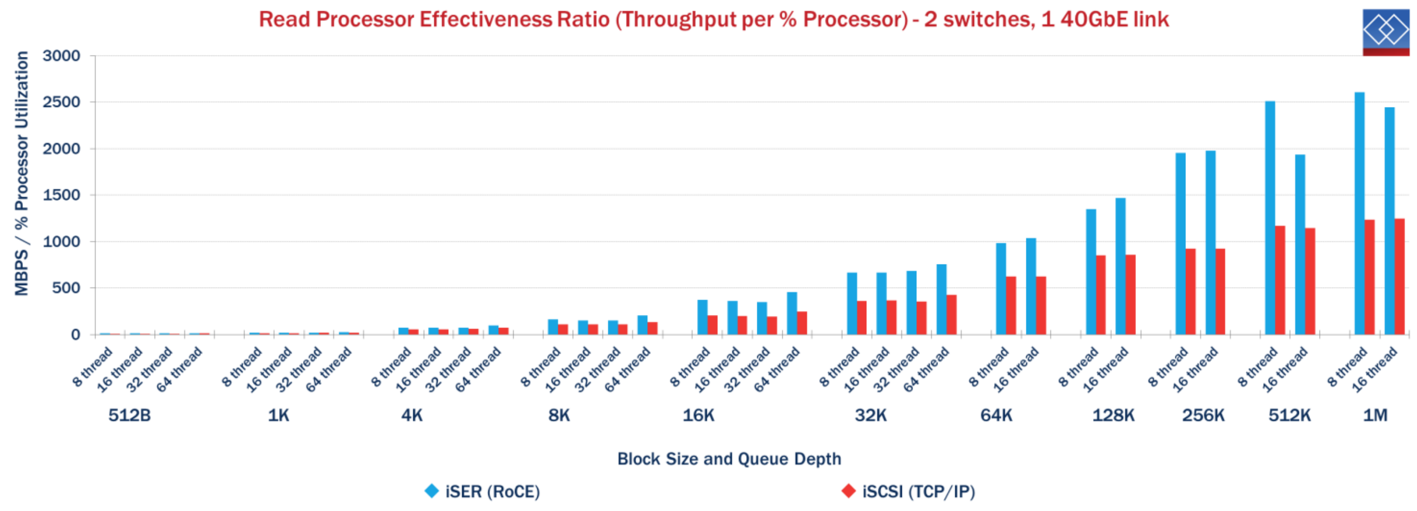


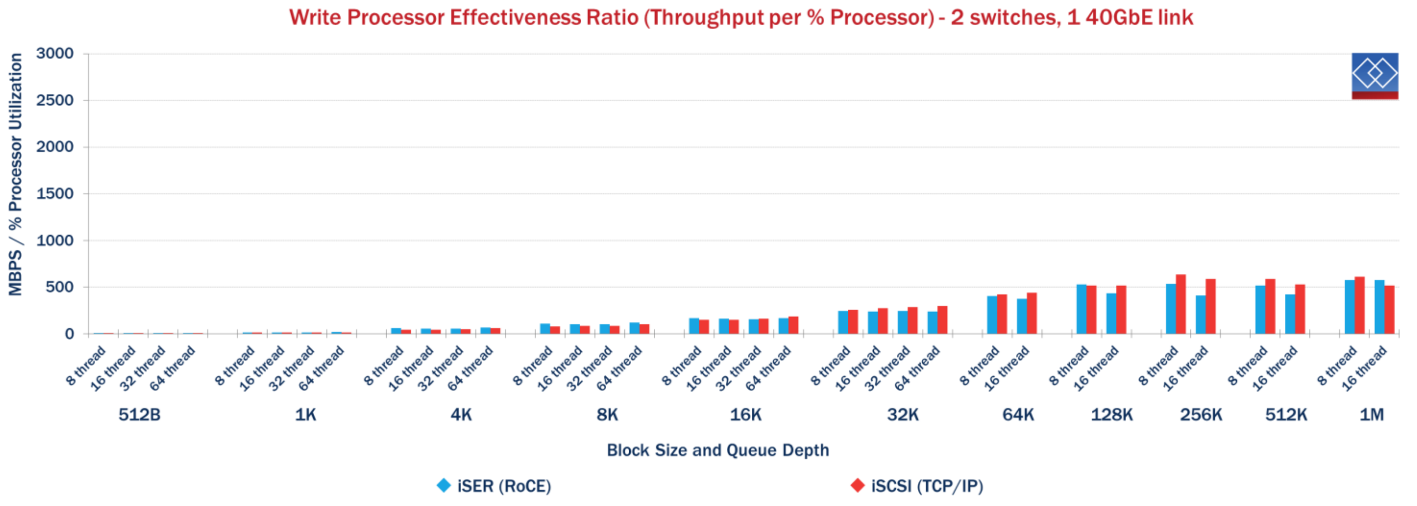
However, it would appear that our write workload for iSER istruggles to reach line rate while iSCSI does not. It is unclear what is causing this.



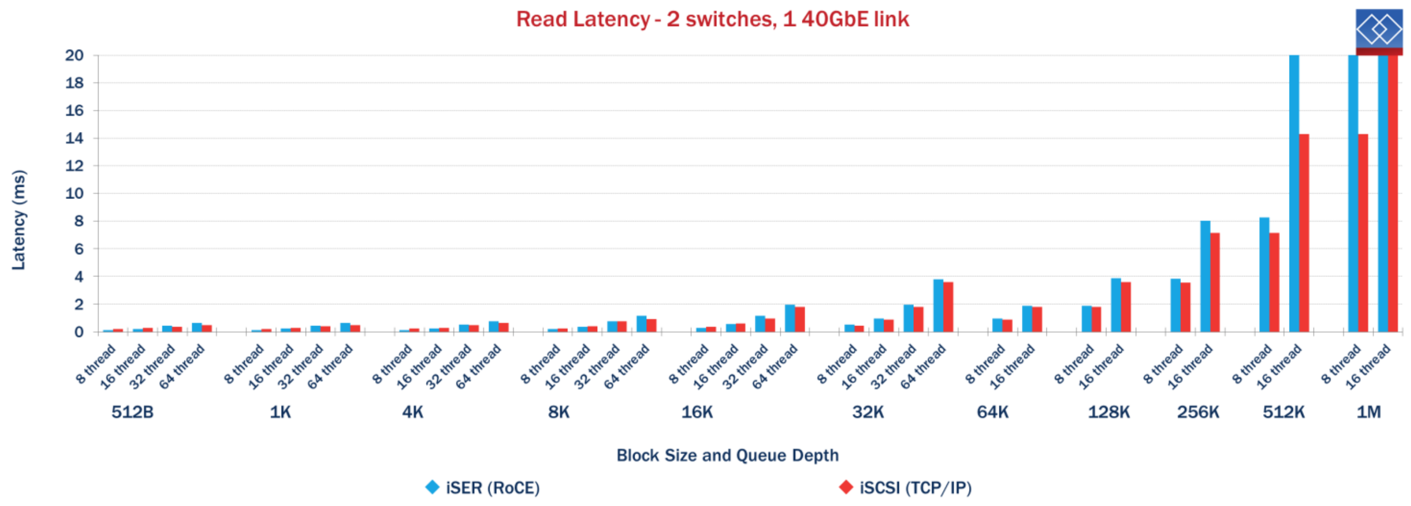


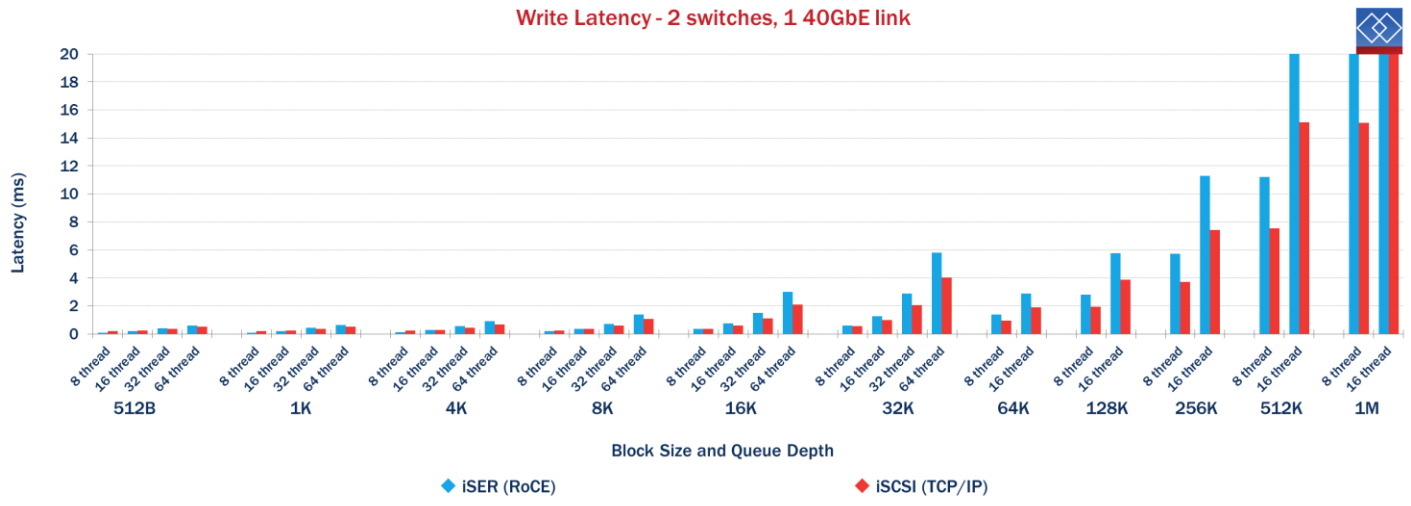
In both examples, processor consumption for iSCSI is consistantly higher than processor consuption for iSER. However with the slightly lower throughput of iSER, we will have to look at processor effectiveness.





Here we see that read processor effectiveness is clearly, consistantly higher for iSER. For writes, processor effectiveness appears to be approximately equal for both protocols.



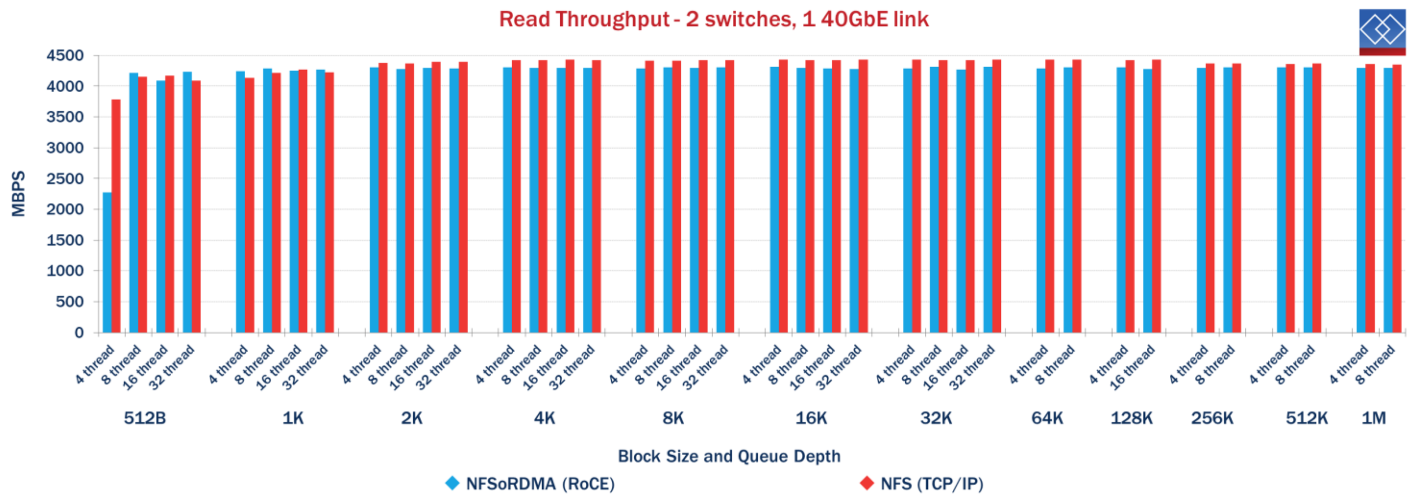


Latencies for both protocols both appear overly high for the workload and target. While we have demonstrated the effectiveness of iSER for reducing processor consumption, it appears we need additional tuning to take advantage of our ramdisk targets.

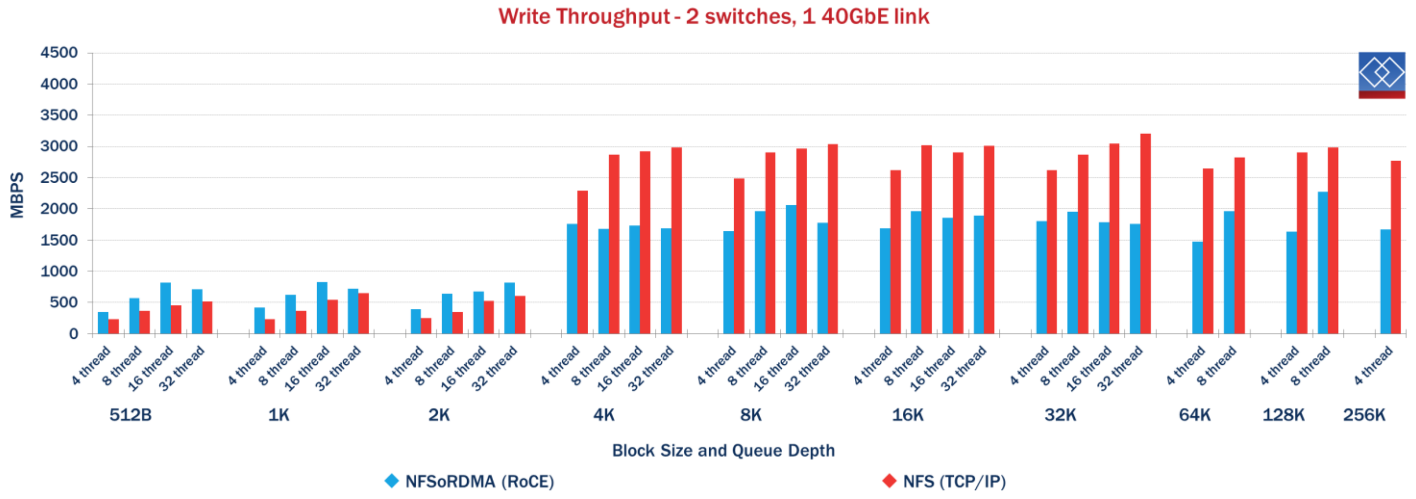
Linux NFSoRDMA with vdbench

The same three server setup that was used for iSER was used for NFSoRDMA testing. The only difference is that attempting to use a high-performance setup where each client port had a separate NFS share led NFSoRDMA to stop working. When an ls was performed on an NFS mountpoint after I/O ceased, the command would hang. Because of this, only one share was created for each client, and only one port on each client was used for NFS.

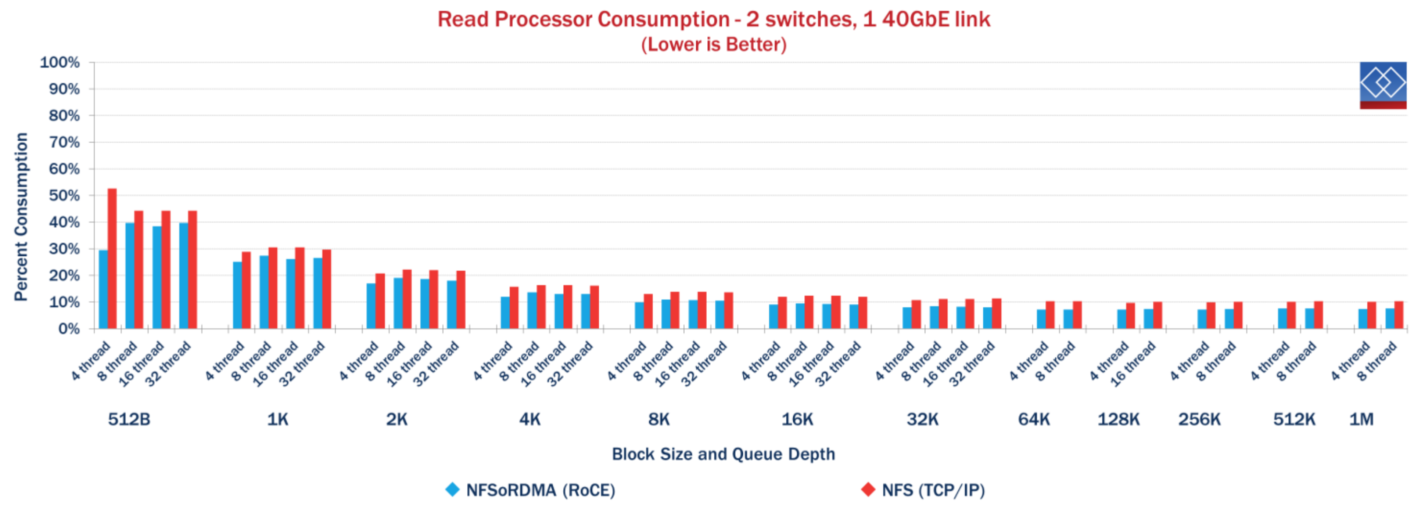
Bear in mind that 2 clients were used, each with a 25GbE connection (50Gbps total), so our maximum throughput is limited as before by the 40GbE switch connection, not the use of one port per NFS client.



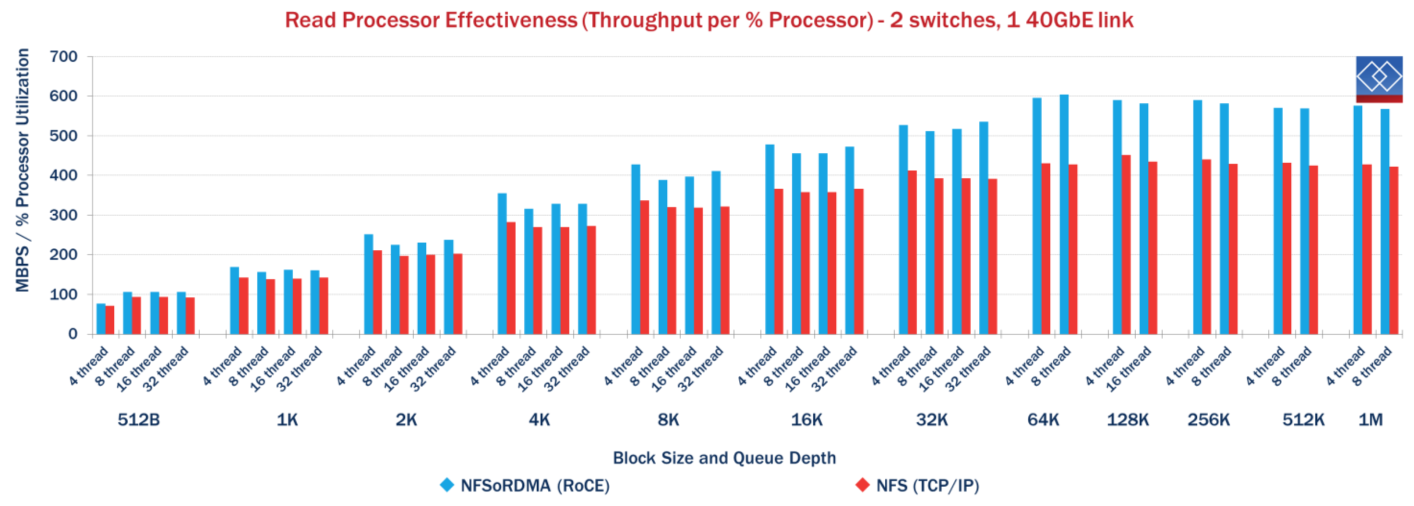
NFSoRDMA and NFS over TCP/IP both appear to easily reach line rate with our read workload



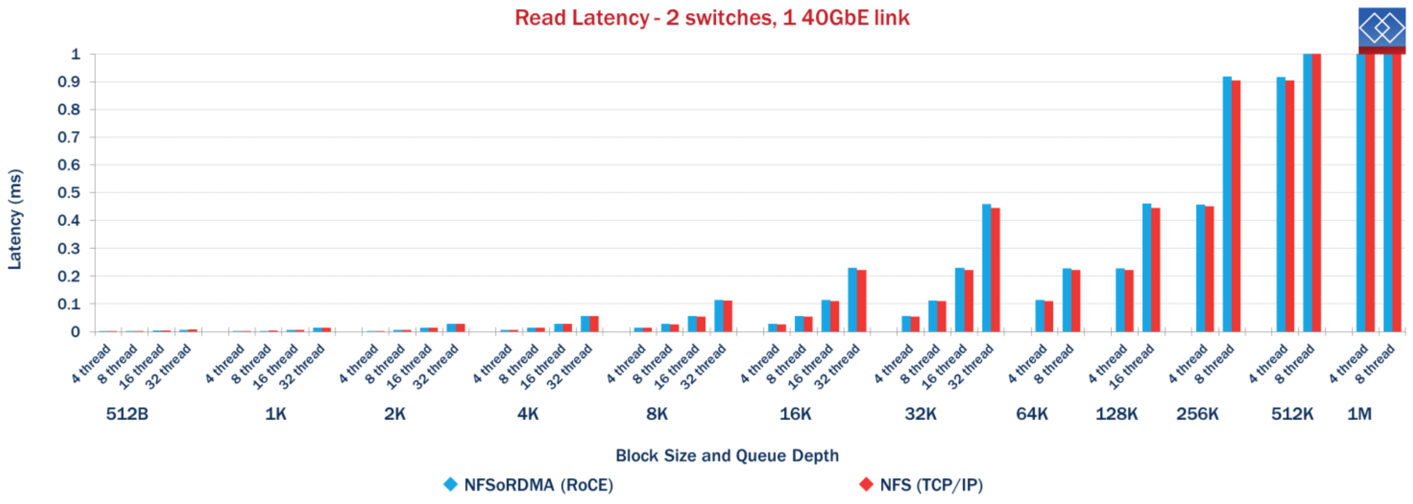
However, our write workload is not reaching line rate. As we are using ramdisk targets, our storage should not be the problem. It is likely something in NFS requires tuning to eliminate the mechanism that might be causing this. We will focus now on the read workload that reached line rate.



We see that read processor for NFS over TCP/IP is consistantly higher than read processor for NFSoRDMA. However, since throughput is also higher it is unclear whether we are gaining benefit from RoCE until we look at the processor effectiveness.



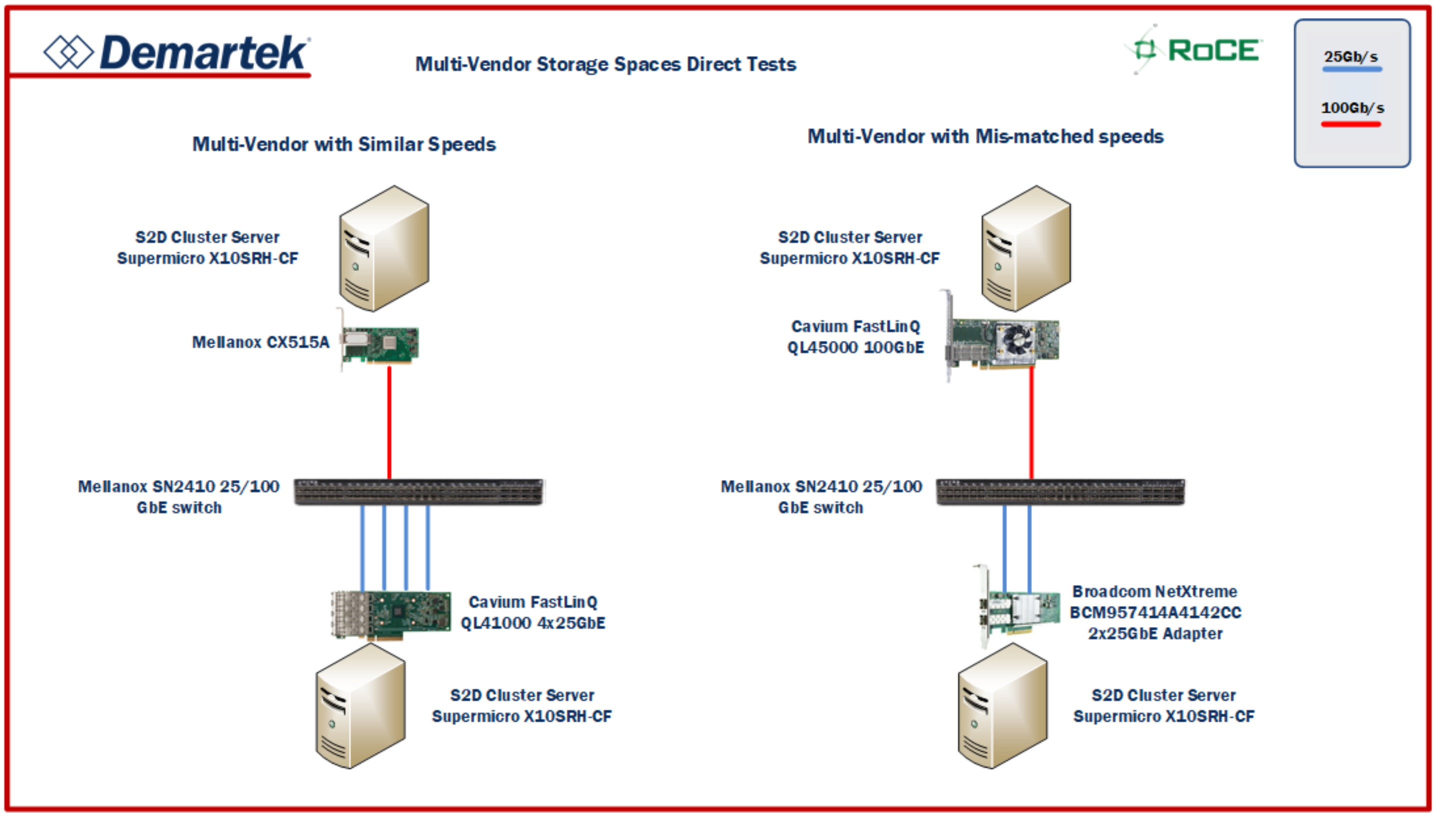
Processor effectiveness for reads is clearly higher for NFSoRDMA RoCE than it is for NFS over TCP/IP, showing the clear benefit of bypassing the TCP/IP stack.



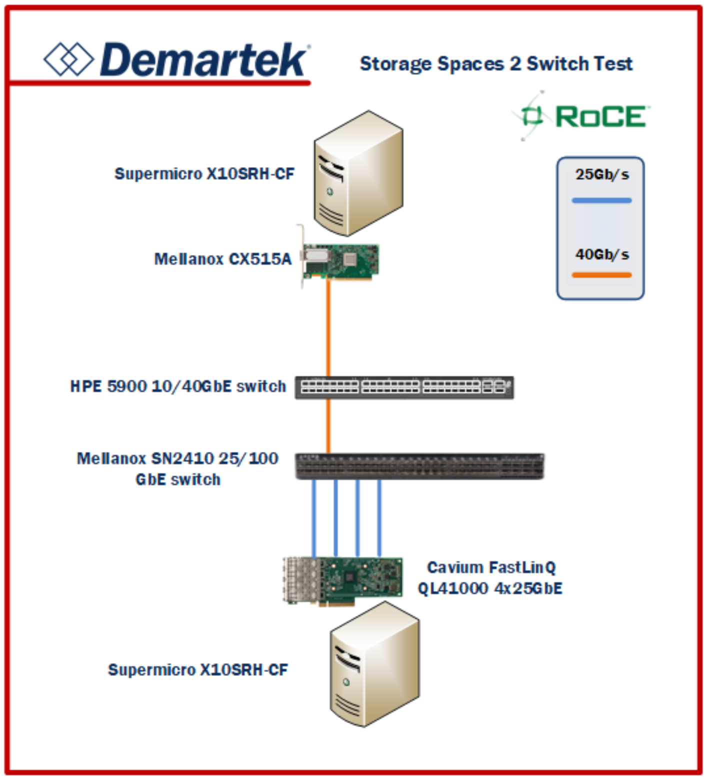
Latencies seem to be low for both RoCE and TCP/IP.

RoCE Configuration Diagrams

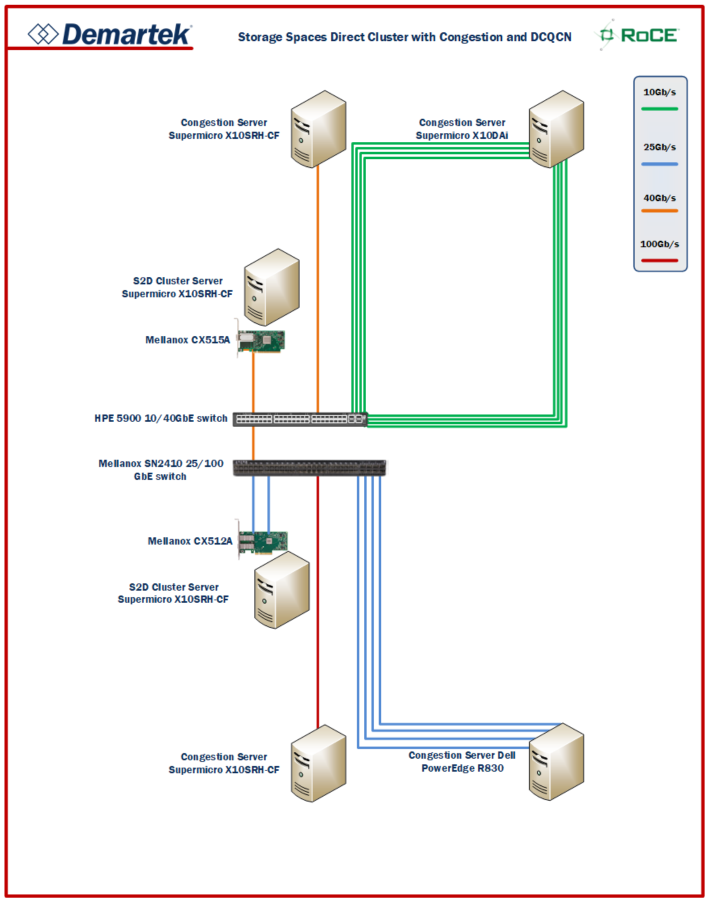
Configurations 1 and 2: Used for Windows Storage Spaces Direct with VMFleet Mixed Vendor Configurations


Demartek
Multi-Vendor Storage Spaces Direct Tests
RoCE
25Gb/s
100Gb/s
Multi-Vendor with Similar Speeds
Multi-Vendor with Mis-matched speeds
S2D Cluster Server Supermicro X10SRH-CF
Mellanox CX515A
Mellanox SN2410 25/100 GbE switch
Cavium FastLinQ QL41000 4x25GbE
S2D Cluster Server Supermicro X10SRH-CF
S2D Cluster Server Supermicro X10SRH-CF
Cavium FastLinQ QL45000 100GbE
Mellanox SN2410 25/100 GbE switch
Broadcom NetXtreme BCM957414A4142CC 2x25GbE Adapter
S2D Cluster Server Supermicro X10SRH-CF

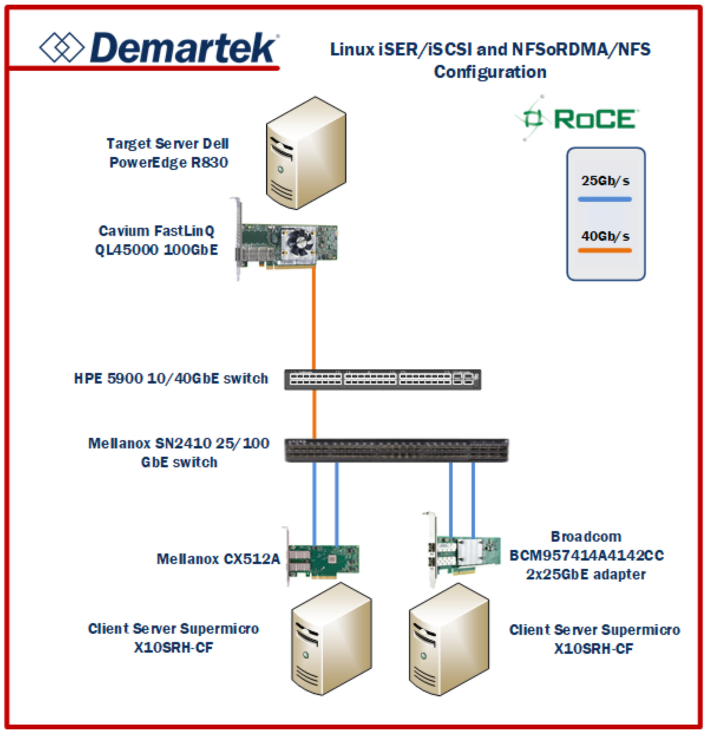
Configuration 3: Used for Windows Storage Spaces with Iometer



Configuration 4: Used for Windows Storage Spaces Direct with VMFleet for The Effects of ETS and RoCEv2 Congestion Management on Congestion


Demartek
Storage Spaces Direct Cluster with Congestion and DCQCN
RoCE
10Gb/s
25Gb/s
40Gb/s
100Gb/s
Congestion Server Supermicro X10SRH-CF
Congestion Server Supermicro X10DAi
S2D Cluster Server Supermicro X10SRH-CF
Mellanox CX515A
HPE 5900 10/40GbE switch
Mellanox SN2410 25/100 GbE switch
Mellanox CX512A
S2D Cluster Server Supermicro X10SRH-CF
Congestion Server Supermicro X10SRH-CF
Congestion Server Dell PowerEdge R830

Configuration 5: Used for Linux tests of iSER/iSCSI and NFSoRDMA/NFS


Demartek
Linux iSER/iSCSI and NFSoRDMA/NFS Configuration
RoCE
25Gb/s
40Gb/s
Target Server Dell PowerEdge R830
Cavium FastLinQ QL45000 100GbE
HPE 5900 10/40GbE switch
Mellanox SN2410 25/100 GbE switch
Mellanox CX512A
Broadcom BCM957414A4142CC 2x25GbE adapter
Client Server Supermicro X10SRH-CF
Client Server Supermicro X10SRH-CF

Servers

Supermicro Servers X10SRH-CF

- 1x Intel Xeon E5-1650 v3 3.5GHz 6 core
- 64 GB RAM
- 3x Samsung SM-1715 1.6 GB NVMe SSD
- 1x Intel SSD DC P3700 745GB NVMe

Dell PowerEdge R830 Server

- 4x Intel Xeon E5-4669 v4, 2.2GHz, 88 total cores
- 1536 GB RAM

Supermicro X10DAi Server

- 2x Intel Xeon E5-2620 v3, 2.4GHz, 12 total cores
- 64 GB RAM

Switches

Mellanox SN2410 25/100GbE Switch

- 48 ports 25GbE
- 8 ports 100GbE

HPE 5900 Series JG838A 10/40GbE Switch

- 48 ports 10GbE
- 4 ports 40GbE

Adapters

Mellanox

- ConnectX-5 MCX512A dual-port 25GbE
- ConnectX-5 MX515A single-port 100GbE

Cavium

- FastLinQ QL41234HLCU series quad-port 25GbE
- FastLinQ QL45611HLCU series single-port 1x100GbE/4x25GbE

Broadcom

- NetXtreme BCM957414A4142CC 2x25GbE adapters

Storage Systems

HPE Storage

- HPE 3PAR StoreServ File Controller
- HPE 3PAR StoreServ 8450 All-Flash Storage

Conclusion

Generally, we found that using RoCE technology provided higher processor effectiveness than TCP/IP. On a small or highly utilized server the use of RoCE can help avoid the performance cap that might be seen with TCP/IP due to the decreased CPU utilization of RoCE.

The addition of ETS and RoCEv2 Congestion Management resulted in improved RoCE throughput and decreased latency.

Multi-vendor configurations with RoCE can perform well in both Windows and Linux environments.

Having knowledge of Ethernet switch commands is beneficial to deploying RoCE solutions, and these commands may vary by switch vendor.

Deploying RoCE technology can boost the performance or lower the CPU consumption of applications that use Ethernet networks.

The most current version of this report is available at https://www.demartek.com/Demartek_RoCE_Deployment_Guide.html on the Demartek website.

Broadcom and NetXtreme are registered trademarks of Broadcom, Inc.

Cavium, the Cavium logo and FastLinQ are trademarks or registered trademarks of Cavium Corporation.

Mellanox, Mellanox logo and ConnectX are registered trademarks of Mellanox Technologies, Ltd.

Demartek is a registered trademark of Demartek, LLC.

All other trademarks are the property of their respective owners.