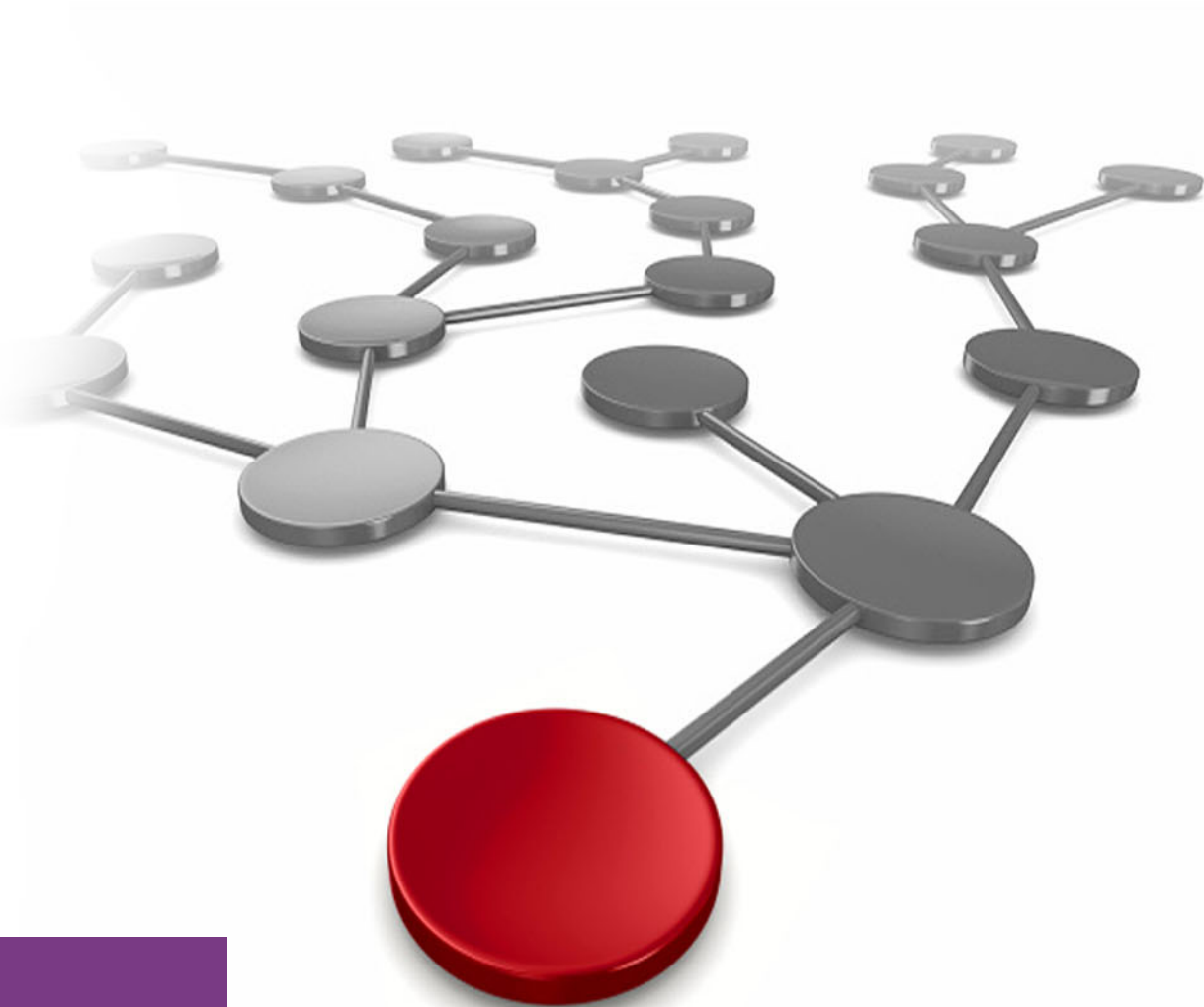


# Highly Efficient Data Access with RoCE on IBM Elastic Storage Systems and IBM Spectrum Scale

Olaf Weiser

Gero Schmidt

Piyush Chaudhary



Storage





IBM Redbooks

**Highly Efficient Data Access with RoCE on IBM ESSs  
and IBM Spectrum Scale**

February 2022

**Note:** Before using this information and the product it supports, read the information in “Notices” on page v.

**First Edition (February 2022)**

This edition applies to IBM Elastic Storage Server models 3000, 3200, and 5000, Version 5, Release 1, Modification 0.2 or later of IBM Spectrum Scale, and Red Hat Enterprise Linux 8.x or later.

**© Copyright International Business Machines Corporation 2022. All rights reserved.**

Note to U.S. Government Users Restricted Rights -- Use, duplication or disclosure restricted by GSA ADP Schedule Contract with IBM Corp.

# Contents

<b>Notices</b> .....	v
Trademarks .....	vi
<b>Preface</b> .....	vii
Authors .....	vii
Now you can become a published author, too! .....	viii
Comments welcome .....	viii
Stay connected to IBM Redbooks .....	ix
<b>Summary of changes</b> .....	xi
February 2022, First Edition (minor updates) .....	xi
<b>Chapter 1. Overview</b> .....	1
<b>Chapter 2. IBM Spectrum Scale Support Matrix</b> .....	3
<b>Chapter 3. Remote Direct Memory Access over Converged Ethernet setup</b> .....	5
3.1 Network requirements .....	6
3.2 Network topology .....	6
3.3 Connection Manager and Subnet Manager .....	8
3.4 Host configuration .....	9
3.4.1 Operating system setting for AMD hardware .....	10
3.4.2 Optional: Ring buffer and other adapter settings .....	10
3.4.3 Maximum Transmission Unit consideration .....	11
3.4.4 Firewall .....	11
3.4.5 Bond configuration .....	11
3.4.6 Regular IP interface configuration .....	13
3.4.7 Routing configurations .....	16
3.4.8 The sysctl settings .....	20
3.4.9 Applying interface settings for Differentiated Services Code Point and Priority Frame Control .....	20
<b>Chapter 4. Functional verification of the network</b> .....	23
4.1 Quick test: ping .....	24
4.2 Quick test: ib_read_bw .....	25
4.3 Nsdperf test with TCP .....	26
4.4 The nsdperf command with Remote Direct Memory Access .....	28
4.5 Debugging method for a network by using system on-board tools .....	30
<b>Chapter 5. IBM Spectrum Scale settings for Remote Direct Memory Access over Converged Ethernet</b> .....	33
<b>Chapter 6. Functional verification of IBM Spectrum Scale</b> .....	35
<b>Appendix A. Mellanox switch configuration for Remote Direct Memory Access over Converged Ethernet</b> .....	37
<b>Appendix B. Configuring Multi-Chassis Link Aggregation on Mellanox switches</b> .....	39
<b>Related publications</b> .....	41
IBM Redbooks .....	41

Online resources .....	41
Help from IBM .....	42
<b>Abbreviations and acronyms .....</b>	<b>43</b>

# Notices

This information was developed for products and services offered in the US. This material might be available from IBM in other languages. However, you may be required to own a copy of the product or product version in that language in order to access it.

IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not grant you any license to these patents. You can send license inquiries, in writing, to:

*IBM Director of Licensing, IBM Corporation, North Castle Drive, MD-NC119, Armonk, NY 10504-1785, US*

INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some jurisdictions do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Any references in this information to non-IBM websites are provided for convenience only and do not in any manner serve as an endorsement of those websites. The materials at those websites are not part of the materials for this IBM product and use of those websites is at your own risk.

IBM may use or distribute any of the information you provide in any way it believes appropriate without incurring any obligation to you.

The performance data and client examples cited are presented for illustrative purposes only. Actual performance results may vary depending on specific configurations and operating conditions.

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

Statements regarding IBM's future direction or intent are subject to change or withdrawal without notice, and represent goals and objectives only.

This information contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to actual people or business enterprises is entirely coincidental.

## COPYRIGHT LICENSE:

This information contains sample application programs in source language, which illustrate programming techniques on various operating platforms. You may copy, modify, and distribute these sample programs in any form without payment to IBM, for the purposes of developing, using, marketing or distributing application programs conforming to the application programming interface for the operating platform for which the sample programs are written. These examples have not been thoroughly tested under all conditions. IBM, therefore, cannot guarantee or imply reliability, serviceability, or function of these programs. The sample programs are provided "AS IS", without warranty of any kind. IBM shall not be liable for any damages arising out of your use of the sample programs.

# Trademarks

IBM, the IBM logo, and ibm.com are trademarks or registered trademarks of International Business Machines Corporation, registered in many jurisdictions worldwide. Other product and service names might be trademarks of IBM or other companies. A current list of IBM trademarks is available on the web at “Copyright and trademark information” at <http://www.ibm.com/legal/copytrade.shtml>

The following terms are trademarks or registered trademarks of International Business Machines Corporation, and might also be trademarks or registered trademarks in other countries.

DS8000®

IBM®

IBM Elastic Storage®


IBM Research®

IBM Spectrum®

POWER8®

POWER9™

Redbooks®

Redbooks (logo) ®

The following terms are trademarks of other companies:

Intel, Intel logo, Intel Inside logo, and Intel Centrino logo are trademarks or registered trademarks of Intel Corporation or its subsidiaries in the United States and other countries.

The registered trademark Linux® is used pursuant to a sublicense from the Linux Foundation, the exclusive licensee of Linus Torvalds, owner of the mark on a worldwide basis.

OpenShift, Red Hat, are trademarks or registered trademarks of Red Hat, Inc. or its subsidiaries in the United States and other countries.

Other company, product, or service names may be trademarks or service marks of others.

# Preface

With Remote Direct Memory Access (RDMA), you can make a subset of a host's memory directly available to a remote host. RDMA is available on standard Ethernet-based networks by using the RDMA over Converged Ethernet (RoCE) interface. The RoCE network protocol is an industry-standard initiative by the InfiniBand Trade Association.

This IBM® Redpaper publication describes how to set up RoCE to use within an IBM Spectrum® Scale cluster and IBM Elastic Storage® Systems (ESSs).

This book is targeted at technical professionals (consultants, technical support staff, IT Architects, and IT Specialists) who are responsible for delivering cost-effective storage solutions with IBM Spectrum Scale and IBM ESSs.

## Authors

This paper was produced by a team of specialists from around the world working at the IBM Redbooks, Tucson Center.

**Olaf Weiser** joined IBM as an experienced professional more than 10 years ago and worked in the DACH TSS team delivering IBM Power based solutions to enterprise and high-performance computing (HPC) customers. He developed deep skills in IBM Spectrum Scale (previously IBM GPFS) and has a worldwide reputation as the performance-optimization specialist for IBM GPFS. At the IBM European Storage Competence Center (ESCC), Olaf works on Advanced Technical Support (ATS) and Lab Services and Skill Enablement tasks that are required to grow the IBM Spectrum Scale business in EMEA. For the past two years, he worked as a performance engineer for RDMA and RoCE in IBM Research® and Development GmbH.

**Gero Schmidt** is a Software Engineer at IBM Germany Research and Development GmbH in the IBM Spectrum Scale development group. He joined IBM in 2001 by working at the ESCC in Mainz, Germany to provide technical presales support for a broad range of IBM storage products, with a primary focus on enterprise storage solutions, system performance, and IBM Power servers. Gero participated in the product rollout of the IBM System Storage DS6000 and IBM System Storage DS8000® series, co-authored several IBM Redbooks®, and was a frequent speaker at IBM international conferences. In 2015, he joined the storage research group at the IBM Almaden Research Center in California, US, where he worked on IBM Spectrum Scale, compression of genomic data in next generation sequencing pipelines, and the development of a cloud-native backup solution for containerized applications in Kubernetes and Red Hat OpenShift. In 2018, he joined the Big Data & Analytics team in the IBM Spectrum Scale development group. He holds a degree in Physics (Dipl.-Phys.) from the Braunschweig University of Technology in Germany.

**Piyush Chaudhary** is a Senior Technical Staff Member who works as a Big Data and Analytics Architect in the IBM Spectrum Scale development team. He has over 18 years of product architecture and design experience. Piyush has an extensive background in HPC and distributed systems. His current focus is on enabling big data frameworks like Hadoop and Spark on IBM Spectrum Scale.

Thanks to the following people for their contributions to this project:

Larry Coyne  
**IBM Redbooks, Tucson Center**

John Lewars  
**IBM Systems, Storage**

Dale McCurdy  
**IBM Global Markets**

Ralph Wuerthner  
**IBM Research Kelsterbach**

Zoltan A Nagy  
**IBM Research Zurich, Technical Lead - Zurich Compute Cloud**

Matthew Shared  
**NVIDIA (Mellanox)**

## Now you can become a published author, too!

Here's an opportunity to spotlight your skills, grow your career, and become a published author—all at the same time! Join an IBM Redbooks residency project and help write a book in your area of expertise, while honing your experience using leading-edge technologies. Your efforts will help to increase product acceptance and customer satisfaction, as you expand your network of technical contacts and relationships. Residencies run from two to six weeks in length, and you can participate either in person or as a remote resident working from your home base.

Find out more about the residency program, browse the residency index, and apply online at:

[ibm.com/redbooks/residencies.html](http://ibm.com/redbooks/residencies.html)

## Comments welcome

Your comments are important to us!

We want our papers to be as helpful as possible. Send us your comments about this paper or other IBM Redbooks publications in one of the following ways:

- Use the online **Contact us** review Redbooks form found at:

[ibm.com/redbooks](http://ibm.com/redbooks)

- Send your comments in an email to:

[redbooks@us.ibm.com](mailto:redbooks@us.ibm.com)

- Mail your comments to:

IBM Corporation, IBM Redbooks  
Dept. HYTD Mail Station P099  
2455 South Road  
Poughkeepsie, NY 12601-5400

## Stay connected to IBM Redbooks

- ▶ Find us on LinkedIn:  
<http://www.linkedin.com/groups?home=&gid=2130806>
- ▶ Explore new Redbooks publications, residencies, and workshops with the IBM Redbooks weekly newsletter:  
<https://www.redbooks.ibm.com/Redbooks.nsf/subscribe?OpenForm>
- ▶ Stay current on recent Redbooks publications with RSS Feeds:  
<http://www.redbooks.ibm.com/rss.html>



# Summary of changes

This section describes the technical changes made in this edition of the paper and in previous editions. This edition might also include minor corrections and editorial changes that are not identified.

Summary of Changes  
for Highly Efficient Data Access with RoCE on IBM ESSs and IBM Spectrum Scale  
as created or updated on February 17, 2022.

## February 2022, First Edition (minor updates)

This revision includes updates from reader feedback. Change bars are turned on to highlight statements that were added or modified.





# Overview

With Remote Direct Memory Access (RDMA), you can a subset of host's memory directly available to a remote host. RDMA is available on standard Ethernet-based networks by using the RDMA over Converged Ethernet (RoCE) interface. The RoCE network protocol is an industry-standard initiative by the InfiniBand Trade Association.

After working with the Mellanox team to qualify the latest RoCE software and firmware with IBM Spectrum Scale and IBM Elastic Storage Systems (ESSs), IBM can provide a powerful solution that uses RoCE V2 for a scale-out network topology that uses high-speed storage tier machines.

This paper describes how to set up RoCE to use within an IBM Spectrum Scale cluster and IBM Elastic Storage Systems (ESSs).

With the current state of the OpenFabrics Enterprise Distribution (OFED) and firmware driver, you can optimize for performance and latency and simultaneously for better availability. OFED supports individual ports, multiple fabrics, and classical network bonds to protect a service IP from network failures.





# IBM Spectrum Scale Support Matrix

The most important starting point to get the latest information about supported software and firmware levels and more information about the facts and features of IBM Spectrum Scale is the FAQ page at [IBM Documentation](#).

IBM Spectrum Scale supports Remote Direct Memory Access (RDMA) on Linux only. IBM Spectrum Scale uses the VERBS programming interface to provide RDMA support. Although IBM Spectrum Scale uses the VERBS programming interface for RDMA support, the underlying implementation of RDMA is vendor-specific.

IBM Spectrum Scale supports RDMA with RDMA over Converged Ethernet (RoCE) V2 with the following configurations:

- ▶ Supported on Linux only.
- ▶ Supported on the x86\_64 and ppc64le platforms.
- ▶ The minimum required Mellanox OpenFabrics Enterprise Distribution (MOFED) is Level 4.9 or later.
- ▶ The minimum IBM Spectrum Scale version is Version 5.1.2.0 or later.
- ▶ The minimum IBM Elastic Storage System (ESS) models are ESS5x and ESS3x.
- ▶ The minimum ESS release is ESS V6.1.2.0.
- ▶ The minimum supported operating system (OS) levels are:
  - Red Hat Enterprise Linux (RHEL) 8.x or later
  - Ubuntu 18.04 or later





## Remote Direct Memory Access over Converged Ethernet setup

In general, there are two different ways to deploy an RDMA over Converged Ethernet (RoCE) environment. The simplest way is to have one adapter port per node that is connected to the network that uses a TCP daemon for communication and uses the same port for RoCE traffic concurrently.

The more complex but powerful and flexible configuration is to have multiple ports that are connected to the network that use one port for TCP daemon communication and Remote Direct Memory Access (RDMA) in parallel simultaneously and all the other ports for only RDMA and RoCE traffic. With this configuration, a node's bandwidth capabilities can scale out nearly linearly.

Optionally, Mellanox and IBM have developed an option to configure a network bond, which consists of two ports from the same adapter to protect against cable, port, or switch issues.

For best performance results and reliability, it is a best practice to have a network that is configured to be "lossless". This RoCE environment is summarized in this chapter.

## 3.1 Network requirements

For running RoCE, the best performance that you can get is when the network is configured as a “lossless” fabric. Depending on the vendor, components, and the topology, this “lossless” requirement can lead to a complex configuration that is out of scope of this document. However, you can find an example configuration in Appendix A, “Mellanox switch configuration for Remote Direct Memory Access over Converged Ethernet” on page 37.

To have a lossless fabric configured, consult your network administration and provider. For more information, see [Recommended Network Configuration Examples for RoCE Deployment](#).

Make sure that all the nodes in your cluster have the most recent Mellanox OpenFabrics Enterprise Distribution (MOFED) driver that is installed properly. The IBM Elastic Storage System (ESS) I/O server nodes are maintained by IBM Spectrum Scale deployment. You can check the installed version by running the `ofed_info -s` command:

```
[root@c902ess1-gssio1 network-scripts]# ofed_info -s
MLNX_OFED_LINUX-x.x.x.x.x
[root@c902ess1-gssio1 network-scripts]#
```

The minimum required level for OpenFabrics Enterprise Distribution (OFED) is documented within the release notes of the ESS.

It is a best practice that the IBM Spectrum Scale client nodes run the same MOFED level as the Network Shared Disk (NSD) or ESS.

However, in many projects and environments, it is a hard to maintain all nodes with the same MOFED level. We have seen successful examples in the market where the client nodes running the OFED are distributed within the operating system (OS), which more convenient for operating and maintaining the client clusters. However, in cases of issues and network glitches, those configurations can cause unexpected failures.

## 3.2 Network topology

The term *host* in this chapter is used for an endpoint in the network. It can be an IBM Spectrum Scale client machine or an ESS or NSD server machine.

A host's configuration depends on the number of network ports that are used or needed. The number of network ports that a node should be connected to in the network depends on the expected bandwidth or on the number of different networks the host must have access to in other environments. The number of ports also is determined by high availability (HA) requirements.

In TCP environments, scaling bandwidth with multiple ports is acquired by using *bonding* network ports, which are also known as *link aggregation*. But, bonding has some challenging side effects, and it makes the deployment more complex from the network perspective.

Bonding is a commonly used technology in data centers to better load balance traffic in the network. However, bonding does not help single-socket network traffic to use more than one cable. Furthermore, another significant drawback of bonding is that the selection of the network path is hard to predict or cannot be controlled from the application layer.

The biggest negative side effect of using network bonds is that Inter-switch Links (ISLs) between switches are used, which adds more network hops and latency.

So, when scaling bandwidth over multiple ports is needed, RoCE is the better option because it saves many system and network resources by using RDMA because the application directly accesses the data through the network without using the OS's resources. Independently from using link aggregation, when you use IBM Spectrum Scale, the generic rule is to have exactly one IP interface that is connected to the IP range for the mmfs daemon-to-daemon communication. A best practice network topology with bonding and RoCE looks like Figure 3-1.

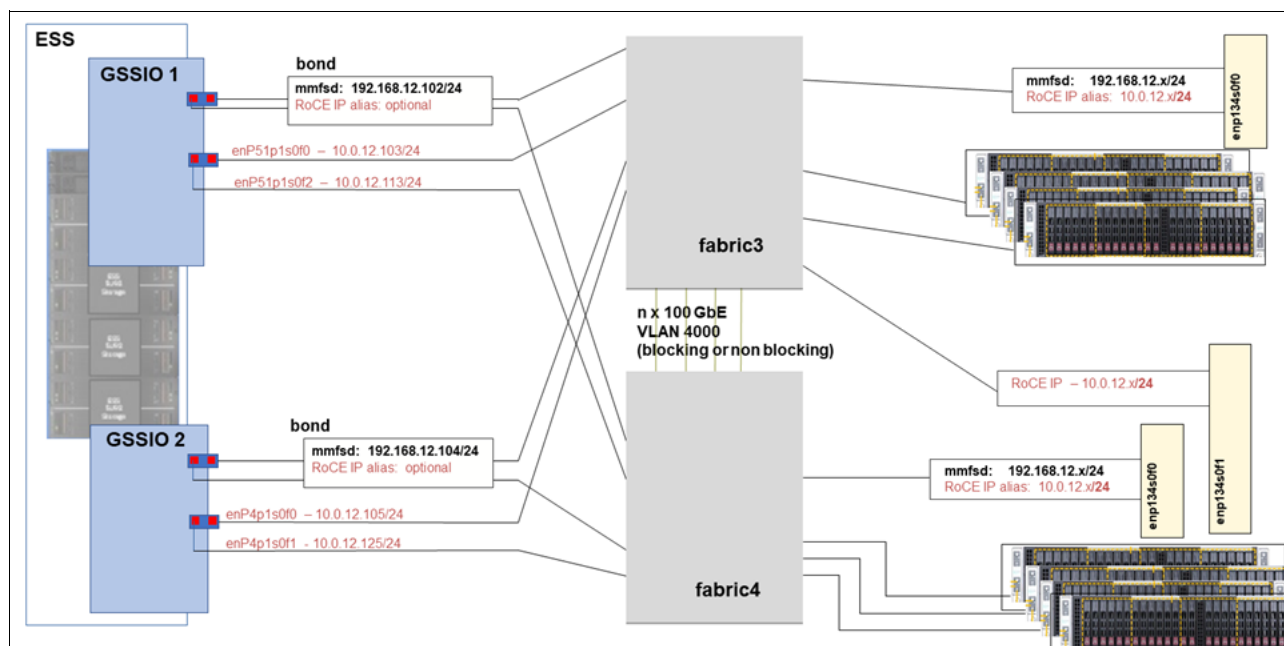


Figure 3-1 Network topology with bonding for mmfsd communication

When using a bond, you must set the Link Aggregation Control Protocol (LACP) link aggregation in the network.

Alternatively, by using ESS building blocks, you can also rely on higher HA layers in IBM Spectrum Scale like NSD and Recovery-Group server failover, so you might consider skipping bonds in your topology to make the network setup less complex. For improved performance and a less complex setup, a best practice is to use a configuration without bonding.

An example for a configuration without bonding looks like Figure 3-2.

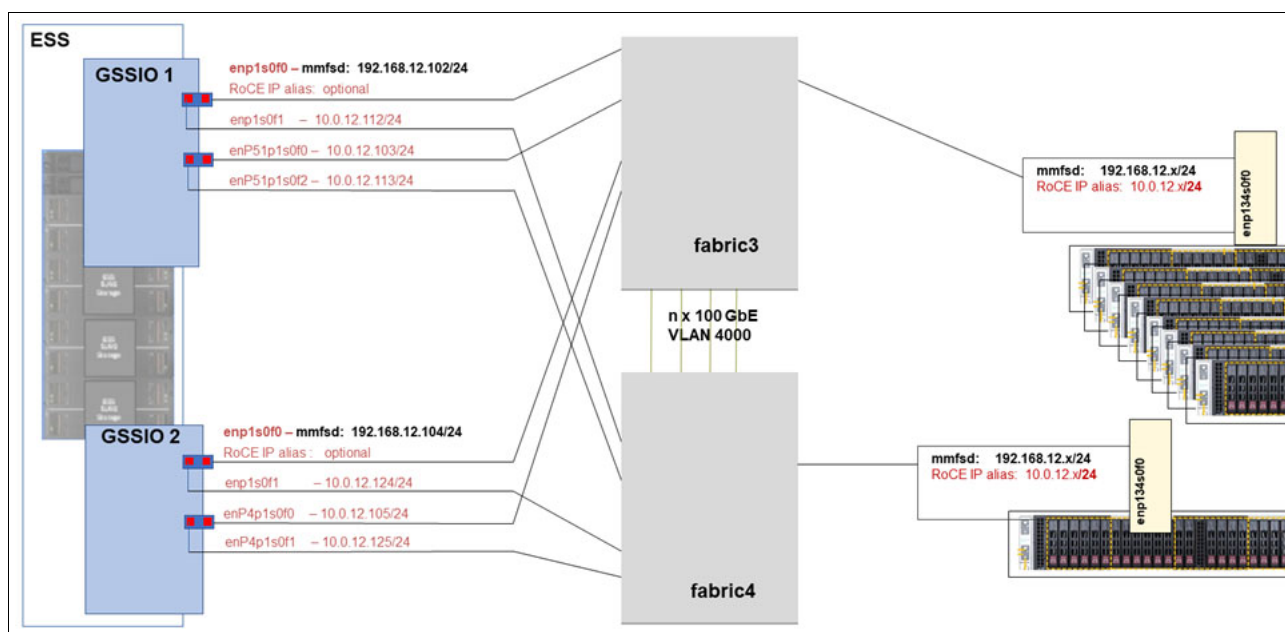


Figure 3-2 Network topology without bonding

In a network topology without bonding, you must have one IP address per network port. On the adapter that has the mmfsd IP address configured, you do not need an extra RoCE IP address or alias if all the nodes in your cluster can communicate to this mmfsd IP address. You can configure as many aliases as the OS version supports. You need one IP per adapter, and RoCE also can use the existing IP that also is used for TCP/IP traffic.

By running a configuration without bonds in the IBM Spectrum Scale configuration, you can enhance the whole configuration by adding fabric numbers. Therefore, traffic on the ISLs can be avoided, and performance can be optimized.

**Note:** The adapters in a RoCE-enabled environment can run TCP/IP traffic and RDMA traffic simultaneously.

### 3.3 Connection Manager and Subnet Manager

If you operate an InfiniBand network, the Subnet Manager (SM) often is used. The SM is a real service, running mostly on the switches, the hosts, or both. The SM discovers and configures all the devices in an InfiniBand fabric. When using the SM in an InfiniBand fabric, you do not have to configure the IP address on the hosts or switches.

The Connection Manager (CM) is a special method to identify and set up the connections between communication partners. It can be used for Ethernet and InfiniBand, it is part of the OFED, and it provides an RDMA transport-neutral interface for establishing connections. It is not a central component (that is, running on the switches like SM), but a component that runs on each individual node. The API concepts are based on sockets, but they are adapted for queue pair (QP) based semantics. To use the CM, you must have IP addresses that are configured.

For RDMA over Ethernet, using CM is mandatory. For InfiniBand, CM is an optional alternative to SM.

InfiniBand and Ethernet can be used simultaneously. Ethernet-based RDMA communication always requires an IP address to work. For those nodes that are connected to both fabrics, you must configure an IP address on all adapter ports, including all InfiniBand ports.

Make sure that a clear separation of IP subnets is configured to the different adapters and fabrics because the adapter must be able to communicate through an IP address to allow IBV verbs to establish connections through CM. You cannot create connections between Ethernet ports and InfiniBand ports for RDMA. Even though there are special routing scenarios to make these cross-fabric connections work, they work only for a TCP/IP connection.

However, an IBM Spectrum Scale node can create RDMA connections over InfiniBand to cluster A while having RoCE connections to the nodes of cluster B.

**Note:** When using CM, all nodes in the InfiniBand fabric, which must communicate with each other, need an IP address on their InfiniBand interfaces to establish connections. This situation is true for each node with CM configured, even if SM is still running on the fabric.

## 3.4 Host configuration

In general, we distinguish between a client and a server in an IBM Spectrum Scale environment. The recommendations in this section apply to both types of hosts.

To configure the host adapters in our examples, we decide whether we want to have the IP interface protected by link aggregation or use a regular IP interface.

If bonding is used, RDMA restrictions mean that you can bond only two ports in one bond, and both of those ports must be on the same physical RoCE adapter. So, you get HA protection of the IP address.

All the other adapters of the I/O servers appear as individual devices.

Figure 3-1 on page 7 shows the network topology with bonding for mmfsd communication.

**Note:** If you use bonding, the two ports in the bond must have the switch site configured with LACP (Multi-Chassis Link Aggregation (MLAG)).

In our examples, we have two dual-port adapters in the ESS I/O servers. The output of the commands on the ESS I/O servers looks like the following output:

```
[root@ess5kiol ~]# lspci | grep Mell
0000:01:00.0 Ethernet controller: Mellanox Technologies MT28800 Family [ConnectX-5 Ex]
0000:01:00.1 Ethernet controller: Mellanox Technologies MT28800 Family [ConnectX-5 Ex]
0033:01:00.0 Ethernet controller: Mellanox Technologies MT28800 Family [ConnectX-5 Ex]
0033:01:00.1 Ethernet controller: Mellanox Technologies MT28800 Family [ConnectX-5 Ex]
[root@ess5kiol ~]# ibdev2netdev
mlx5_0 port 1 ==> enp1s0f0 (Up)
mlx5_1 port 1 ==> enp1s0f1 (Up)
mlx5_2 port 1 ==> enP51p1s0f0 (Up)
mlx5_3 port 1 ==> enP51p1s0f1 (Up)
[root@ess5kiol ~]#
```

The clients that we are using have only one adapter, and we use only one port per client:

```
[root@fscs-sr650-13]# lspci | grep Mell
86:00.0 Ethernet controller: Mellanox Technologies MT27700 Family [ConnectX-4]
86:00.1 Ethernet controller: Mellanox Technologies MT27700 Family [ConnectX-4]
[root@fscs-sr650-13]# ibdev2netdev
mlx5_0 port 1 ==> enp134s0f0 (Up)
mlx5_1 port 1 ==> enp134s0f1 (Down)
[root@fscs-sr650-13]#
```

### 3.4.1 Operating system setting for AMD hardware

When enabling InfiniBand on AMD64 hardware, `iommu=soft` might be required in the grub boot options to permit allocations greater than 1 GB to the VERBS RDMA device, which might impact performance and CPU utilization.

### 3.4.2 Optional: Ring buffer and other adapter settings

Depending on the hardware, server type, and models of the client/server setup, you might want to adjust the adapter settings. On ESS, you can use the default settings from the deployment procedure.

The clients that we use in our example are fast enough with the default settings. We reach full wire speed with the default adapter settings, so we keep the defaults on the clients.

However, if you do not see the expected network performance, you might want to change the adapter settings in your environment.

If you do not get the wire speed that you want, here are some approaches.

- ▶ Try to set the rx and tx buffers a bit higher, step by step. If you go from 1024 to 8192 in one step, you might impact running workloads, and depending on the available free memory resources, you might crash the system.
- ▶ When you have your optimized setup, configure an udev rule to make it automatically effective.
- ▶ To take advantage of the network adapter capabilities, check the maximum values for tx and rx buffers on the adapter.

Here is an example of checking the maximum values for TX and RX buffers:

```
[root@lbs2gssio1 ~]# ethtool -g enP4p1s0f0
Ring parameters for enP4p1s0f0:
Pre-set maximums:
RX:                8192
RX Mini:           0
RX Jumbo:          0
TX:                8192
Current hardware settings:
RX:                8192
RX Mini:           0
RX Jumbo:          0
TX:                8192
```

As a best practice, create persistent udev rules to adjust the settings automatically. An example of a valid udev rule follows:

```
# cat /etc/udev/rules.d/99-ibm-custom.rules
KERNEL=="ens5f0", RUN+="/sbin/ip link set %k txqueuelen 10000" , RUN+="/sbin/ip
link set %k txqueuelen 10000", RUN+="/sbin/ethtool -G %k rx 8192" ,
RUN+="/sbin/ethtool -G %k tx 8192"

# a more generic one
# cat /etc/udev/rules.d/99-ibm-custom.rules
KERNEL=="enp*", RUN+="/sbin/ip link set %k txqueuelen 10000" , RUN+="/sbin/ip link
set %k txqueuelen 10000", RUN+="/sbin/ethtool -G %k rx 8192" , RUN+="/sbin/ethtool
-G %k tx 8192"
KERNEL=="enP*", RUN+="/sbin/ip link set %k txqueuelen 10000" , RUN+="/sbin/ip link
set %k txqueuelen 10000", RUN+="/sbin/ethtool -G %k rx 8192" , RUN+="/sbin/ethtool
-G %k tx 8192"

# active by
udevadm control --reload-rules
udevadm trigger
```

### 3.4.3 Maximum Transmission Unit consideration

RDMA was introduced on InfiniBand networks. RDMA over InfiniBand supports Maximum Transmission Unit (MTU) sizes of 256 - 4096 bytes.

On Ethernet, you can configure larger MTUs. As a best practice, adjust the MTU so that you use *jumbo frames*, which are 9000 bytes. Adjust this setting on all adapters and all nodes in your clusters.

If you must communicate with outside nodes in remote networks and you cannot be sure that the path through the network supports MTU 9000 end-to-end, make sure that you have MTU Path discovery enabled. MTU path discovery is enabled by default on Linux. Do not forget to set MTU size also on all the switches in the fabric to be able to benefit from jumbo frames.

You can check MTU path discovery by using the following command:

```
[root@fscc-sr650-18 ~]# cat /proc/sys/net/ipv4/ip_no_pmtu_disc
0
[root@fscc-sr650-18 ~]#
```

### 3.4.4 Firewall

There is no special port that is defined for RoCE traffic. However, you need a working TCP/IP connection between both ends. By using RoCE, you aim for high performance with low latency, so using a network where firewalls are in place is not recommended.

An IBM Spectrum Scale cluster is a trusted environment. It is common practice to put the RDMA interfaces into a trusted zone, as shown in the following command:

```
# firewall-cmd --permanent --zone=trusted --add-interface=ens5f0
```

### 3.4.5 Bond configuration

For RoCE V2 to work correctly, all interfaces need an IPv6 local link address and an IPv4 address.

Configuring bond interfaces is optional and not needed to run RoCE. If you want to have your IP interfaces protected against port failures, follow the procedure in this section.

To configure the bond on Red Hat Enterprise Linux (RHEL), use the **nmcli** utility. For more information about this task, see [Configuring Network Bonding](#).

#### **Bonding issues:**

A bonded interface protects against port and cable failures. However, when running RDMA, all ports of the bonded interface must be on the same physical PCI adapter. So, an adapter failure is not covered by this configuration. Furthermore, creating bonds over multiple switches makes an MLAG configuration mandatory in the network, which can cause unbalanced network utilization in the fabric.

Another significant reason not to use bonding when using RoCE is the better performance that you get when you do not use a bond. Using the bond offers only one virtual port for RDMA. In some performance-optimized environments, RDMA ports in IBM Spectrum Scale can be sorted to different fabrics by referring to isolated traffic to avoid ISL bottlenecks. When you use a bond, you cannot predict the traffic, so RDMA fabrics are hard to use or will cause ISL traffic.

Example 3-1 shows an example of configuring a bond on RHEL by using the **nmcli** utility.

#### *Example 3-1 Configuring a bond on RHEL by using the nmcli utility*

```
nmcli connection add type bond con-name bond0 ifname bond0 bond.options \
"mode=802.3ad,miimon=1000,xmit_hash_policy=layer3+4"

nmcli con add type ethernet slave-type bond con-name bond0-p1 ifname enp1s0f0 master bond0
nmcli con add type ethernet slave-type bond con-name bond0-p2 ifname enp1s0f1 master bond0
nmcli connection modify bond0 ipv4.addresses '192.168.12.102/24'
nmcli con mod bond0 ipv4.method static
nmcli con mod bond0 ipv6.addr-gen-mode eui64

nmcli con mod bond0-p1 802-3-ethernet.mtu 9000
nmcli con mod bond0-p1 mtu 9000
nmcli con mod bond0-p2 802-3-ethernet.mtu 9000
nmcli con mod bond0-p2 mtu 9000
nmcli con mod bond0 mtu 9000
```

An appropriate port channel in the network or switch (that uses LACP or MLAG) must be configured.

For RoCE V2 to work correctly, the interfaces need an IPv6 local link address and an IPv4 address. Make sure that the IPv6 local link address is set correctly, which is highlighted in **bold** in Example 3-1.

Make the changes effective and start the interface by running the following command:

```
[root@lbs2gssio1 ~]# # nmcli connection up bond0
[root@lbs2gssio1 ~]#
```

After activating the bond, the MOFED driver changes the device configuration. Instead of seeing both RDMA devices for the adapter with the bonded ports, the `m1x5_bond_0` device is created. The driver may take up to 5 seconds to reconfigure the devices.

```
[root@ess5kio1 ~]# ibdev2netdev
```

```

mlx5_2 port 1 ==> enP51p1s0f0 (Up)
mlx5_3 port 1 ==> enP51p1s0f1 (Up)
mlx5_bond_0 port 1 ==> bond0 (Up)
[root@ess5kiol ~]#

```

As a best practice, protect only the mmfsd IP address with a bond. All other interfaces can be used without a bond. For more configuration examples, see 3.4.6, “Regular IP interface configuration” on page 13. You can run a similar configuration on the other (ESS) nodes and on the clients.

### 3.4.6 Regular IP interface configuration

For RoCE V2 to work correctly in a regular IP interface, all interfaces need an IPv6 local link address and an IPv4 address. Configure all interfaces where you intend to use RoCE communication, as shown in Figure 3-3.

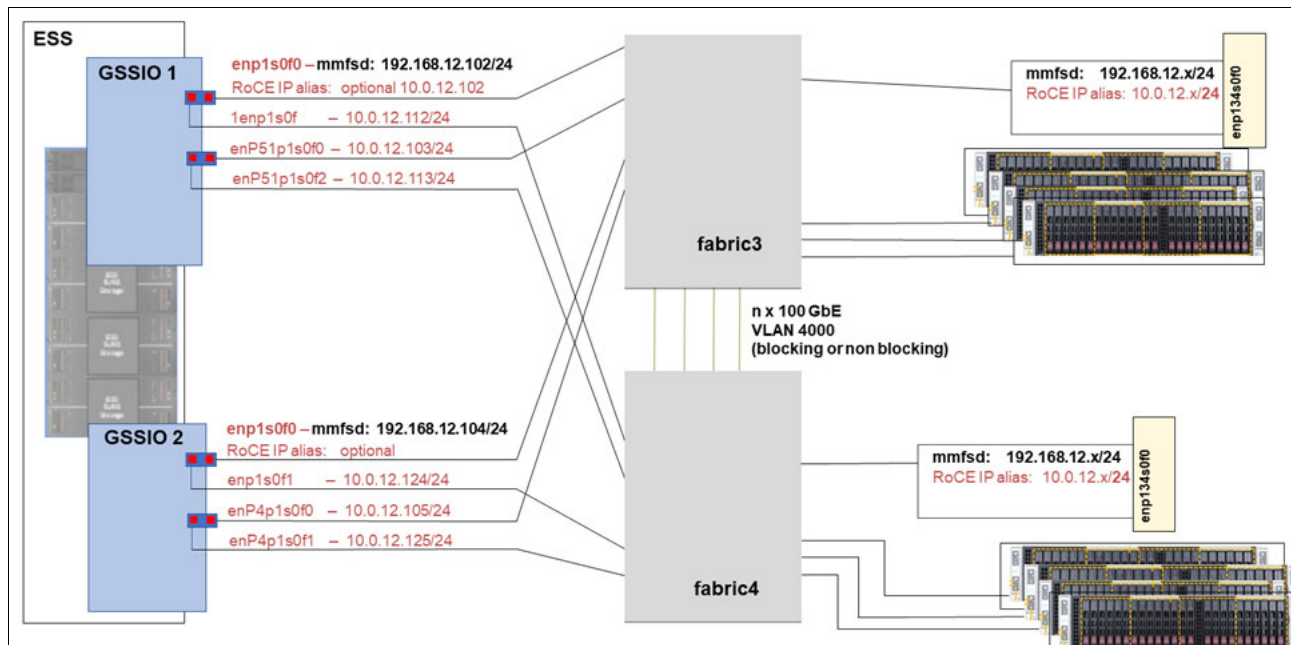


Figure 3-3 Topology without bonding

As highlighted in **bold** in Figure 3-3, make sure that only one IP interface has an IP address for the mmfs daemon communication. All other interfaces must be on a different subnet. Usually, we start configuring the NSD server and then proceed with the clients.

Use the **nmcli** utility to configure your environment, as shown in Example 3-2.

#### Example 3-2 Configuring the environment with the nmcli utility

```

#####
#
# ESS5000 IOservers IO1
#
ifdown enp1s0f0
nmcli con del enp1s0f0
nmcli con add type 802-3-ethernet ifname enp1s0f0 connection.interface-name \
enp1s0f0 connection.id enp1s0f0
nmcli con mod enp1s0f0 ipv4.addresses 192.168.12.102/24

```

```

nmcli con mod enpls0f0 ipv4.method static
nmcli con mod enpls0f0 connection.autoconnect yes
nmcli con mod enpls0f0 802-3-ethernet.mtu 9000
nmcli con mod enpls0f0 ipv6.addr-gen-mode eui64
ifup enpls0f0

ifdown enpls0f1
nmcli con del enpls0f1
nmcli con add type 802-3-ethernet ifname enpls0f1 connection.interface-name enpls0f1 \
connection.id enpls0f1
nmcli con mod enpls0f1 ipv4.addresses 10.0.12.112/24
nmcli con mod enpls0f1 ipv4.method static
nmcli con mod enpls0f1 connection.autoconnect yes
nmcli con mod enpls0f1 802-3-ethernet.mtu 9000
nmcli con mod enpls0f1 ipv6.addr-gen-mode eui64
ifup enpls0f1

ifdown enP51pls0f0
nmcli con del enP51pls0f0
nmcli con add type 802-3-ethernet ifname enP51pls0f0 connection.interface-name enP51pls0f0\
connection.id enP51pls0f0
nmcli con mod enP51pls0f0 ipv4.addresses 10.0.12.103/24
nmcli con mod enP51pls0f0 ipv4.method static
nmcli con mod enP51pls0f0 connection.autoconnect yes
nmcli con mod enP51pls0f0 802-3-ethernet.mtu 9000
nmcli con mod enP51pls0f0 ipv6.addr-gen-mode eui64
ifup enP51pls0f0

ifdown enP51pls0f1
nmcli con del enP51pls0f1
nmcli con add type 802-3-ethernet ifname enP51pls0f1 connection.interface-name enP51pls0f1\
connection.id enP51pls0f1
nmcli con mod enP51pls0f1 ipv4.addresses 10.0.12.113/24
nmcli con mod enP51pls0f1 ipv4.method static
nmcli con mod enP51pls0f1 connection.autoconnect yes
nmcli con mod enP51pls0f1 802-3-ethernet.mtu 9000
nmcli con mod enP51pls0f1 ipv6.addr-gen-mode eui64
ifup enP51pls0f1

```

---

Make sure that the IPv6 local link address is configured correctly by comparing the interface's MAC address with the last fields in the IPv6 address:

```

6: enpls0f0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 9000 qdisc mq state UP group
default qlen 1000
    link/ether 0c:42:a1:d7:1a:e8 brd ff:ff:ff:ff:ff:ff
    inet 10.0.12.102/24 brd 10.0.12.255 scope global noprefixroute enpls0f0
        valid_lft forever preferred_lft forever
    inet 192.168.12.102/24 brd 192.168.12.255 scope global noprefixroute enpls0f0
        valid_lft forever preferred_lft forever
    inet6 fe80::e42:a1ff:fed7:1ae8/64 scope link noprefixroute
        valid_lft forever preferred_lft forever

```

For the I/O server, we have the final IPv4 configuration as shown in the following example:

```

[root@ess5kiol ~]# ip -4 a | grep inet
    inet 127.0.0.1/8 scope host lo

```

```
[...]
inet 10.0.12.102/24 brd 10.0.12.255 scope global noprefixroute enp1s0f0
inet 192.168.12.102/24 brd 192.168.12.255 scope global noprefixroute enp1s0f0
inet 10.0.12.112/24 brd 10.0.12.255 scope global noprefixroute enp1s0f1
inet 10.0.12.103/24 brd 10.0.12.255 scope global noprefixroute enP51p1s0f0
inet 10.0.12.113/24 brd 10.0.12.255 scope global noprefixroute enP51p1s0f1
```

On client machines, which have fewer interfaces than the servers, make sure that the IP interface has all the needed IP aliases to communicate to all server ports, as in the following example:

```
[root@ece-13 ~]# ip -4 a | grep inet
[...]

inet 192.168.12.13/24 brd 192.168.12.255 scope global noprefixroute enp134s0f0
inet 10.0.12.13/24 brd 10.0.12.255 scope global noprefixroute enp134s0f0
```

Regardless of whether you use a bonded or a regular interface, all clients must have an IP address to access the mmfsd daemons network. To enable the client node to connect with the RoCE interfaces to any other nodes with different IP ranges, use either a routing configuration in the network, or as it is demonstrated here in the example, add an IP alias to the interface because it is connected to the same physical network.

The client's network configuration as is shown in the following example:

```
[root@fscs-sr650-13]# ip -4 a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    inet 127.0.0.1/8 scope host lo
        valid_lft forever preferred_lft forever
[...]
6: enp134s0f0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 9000 qdisc mq state UP group default qlen 1000
    inet 192.168.12.13/24 brd 192.168.12.255 scope global noprefixroute enp134s0f0
        valid_lft forever preferred_lft forever
    inet 10.0.12.13/24 brd 10.0.12.255 scope global noprefixroute enp134s0f0
        valid_lft forever preferred_lft forever
```

### 3.4.7 Routing configurations

In our example, the `mmfsd` communication runs in the 192.168.12.0/24 network. All other Mellanox cards interfaces are highlighted in red, shown in Figure 3-4, and they are configured to be in the subnet 10.0.12.x/24. All interfaces are intended to be used for RDMA communication, and only the IP addresses in the 192.168.12.0/24 network are used for TCP/IP communication, which are highlighted in bold in Figure 3-4.

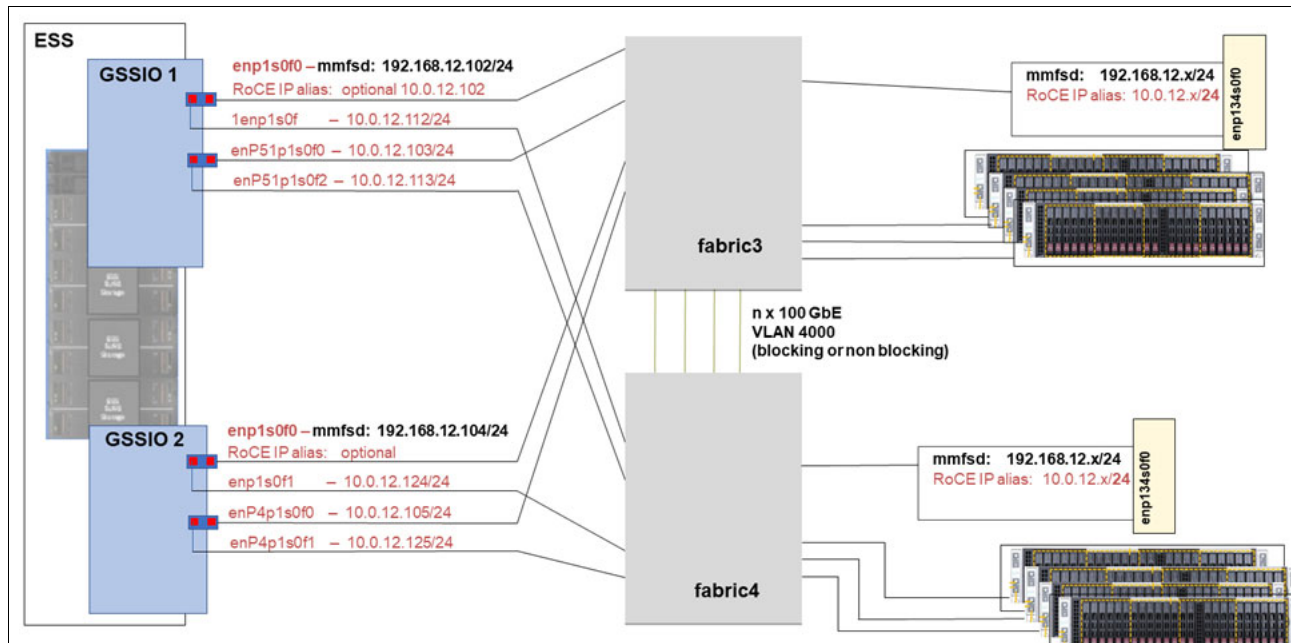


Figure 3-4 Network topology without bonding

With current IBM Spectrum Scale releases, only one IP address per node is supported for TCP/IP communication to the `mmfs` daemon's network.

To scale out over multiple ports, we use RoCE. Per the OFED standard, each RDMA interface must have an IP address to maintain the connection.

Because we want to use multiple interfaces, in theory we must configure multiple subnets, but this configuration is not practical for larger environments and technically not needed. We also can configure all the interfaces that we intend to use for RDMA into one interface that is separate from the `mmfsd` network and subnet.

A limitation of the Linux kernel is that multiple interfaces to the same subnets are challenging. You must consider answering ARP requests, selecting rules for outgoing IP traffic, and other items when you have more than one interface per subnet on a node. To solve those problems, you need to add special routing tables. You must specify only routing entries for each interface and all subnets where you have more than one interface to connect.

In our example, each I/O server node has four interfaces that are connected to the same physical network. For better administration, we assign a 10.0.12.xx/24 address to each interface, and we configure the `mmfsd` IP address on the network port, which is optional because we have a network topology without bonding.

For all interfaces pointing to the 10.0.12.x IP address, an extra routing entry is needed if you have more than one IP interface to the same subnet. If you have only one IP interface to a network, the necessary routing entry for the kernel is done automatically by the OS. With more than one IP interface to the same network, you must create special routes because the Linux kernel does not specifically tie IP addresses to MAC addresses by default. Any network interface controller (NIC) can respond to any ARP request for an IP address that belongs to the server. For RoCE to work correctly, we must make sure that the right interfaces answer connection requests. So, we must create multiple routing entries.

In our example, the clients have only one interface to the servers. The client's single interface has two IP address, one in the daemon network and one for the separated IP range, which can be referenced as a RoCE network. In this configuration, you can skip the optional IP alias for the RoCE network on the server side, which is marked as "optional" in Figure 3-4 on page 16.

If you use clients with multiple adapter ports, the same rule applies to the clients. The clients must have only one IP interface with a mmfsd IP address, but it can have one or many other extra interfaces within the separated IP range (RoCE). For those client interfaces to communicate to all interfaces on the server side, they must have a matching IP address. So, you must configure them as "optional" RoCE IP aliases on the interface with the mmfsd IP address.

To configure the routing, open the `/etc/iproute2/route_tables` file and add one table per interface for all interfaces on the same subnet. In our example, the file contains four extra lines (bolded):

```
[root@c902ess1-gssio1 network-scripts]# cat/etc/iproute2/route_tables
#
# reserved values
#
255      local
254      main
253      default
0        unspec
#
# local
#
#1       inr.ruhep
100 t1
101 t2
102 t3
103 t4
```

Alternatively, you can run the following commands:

```
echo "101 t1" >> /etc/iproute2/route_tables
echo "102 t2" >> /etc/iproute2/route_tables
echo "103 t3" >> /etc/iproute2/route_tables
echo "104 t4" >> /etc/iproute2/route_tables
```

Do not forget to do this task on each node where you have multiple interfaces to the same subnet configured, or distribute this file among your cluster nodes so that they have same configuration.

## Routing entries in RHEL8.x

In RHEL8, the method to set the network configuration was changed in the OS. Setting those routing entries must be done by using the `nmcli` utility. Furthermore, the routing configuration files that were in earlier releases of RHEL 7.x in `/etc/sysconfig/network.scripts/` are no longer present.

To create the routes in RHEL 8.x on the I/O servers, use the script that is shown in Example 3-3.

*Example 3-3 Creating routes on I/O servers in RHEL 8.x*

---

```
nmcli con modify enp1s0f0 +ipv4.routes "0.0.0.0/1 10.0.12.254 table=101, 128.0.0.0/1 10.0.12.254 table=101"
nmcli con modify enp1s0f0 +ipv4.routes "10.0.12.0/24 table=101 src=10.0.12.102"
nmcli con modify enp1s0f0 +ipv4.routing-rules "priority 32761 from 10.0.12.102 table 101"
ifdown enp1s0f0; ifup enp1s0f0

nmcli con modify enp1s0f1 +ipv4.routes "0.0.0.0/1 10.0.12.254 table=102, 128.0.0.0/1 10.0.12.254 table=102"
nmcli con modify enp1s0f1 +ipv4.routes "10.0.12.0/24 table=102 src=10.0.12.112"
nmcli con modify enp1s0f1 +ipv4.routing-rules "priority 32761 from 10.0.12.112 table 102"
ifdown enp1s0f1; ifup enp1s0f1

nmcli con modify enP51pls0f0 +ipv4.routes "0.0.0.0/1 10.0.12.254 table=103, 128.0.0.0/1 10.0.12.254 table=103"
nmcli con modify enP51pls0f0 +ipv4.routes "10.0.12.0/24 table=103 src=10.0.12.103"
nmcli con modify enP51pls0f0 +ipv4.routing-rules "priority 32761 from 10.0.12.103 table 103"
ifdown enP51pls0f0; ifup enP51pls0f0

nmcli con modify enP51pls0f1 +ipv4.routes "0.0.0.0/1 10.0.12.254 table=104, 128.0.0.0/1 10.0.12.254 table=104"
nmcli con modify enP51pls0f1 +ipv4.routes "10.0.12.0/24 table=104 src=10.0.12.113"
nmcli con modify enP51pls0f1 +ipv4.routing-rules "priority 32761 from 10.0.12.113 table 104"
ifdown enP51pls0f1; ifup enP51pls0f1
```

---

For a bond interface that has only the `mmfsd` network IP address, no route is needed because the IP address is the only IP interface into that network. If you prefer to have the optional “red” RoCE address as an extra alias, then a route for the “red” IP subnet also is needed.

In the first line of Example 3-3, a routing entry for one interface (here `enp1s0f0`) is generated for the routing table number 101. It is a default rule that starts from `0.0.0.0/1` and `128.0.0.0/1`. The reason for this layout is that the `nmcli` utility does not support using `0.0.0.0/0` for the default gateway in the field `ipv4.gateway`. To work around this problem, the command creates separate routes for both the `0.0.0.0/1` and `128.0.0.0/1` subnets, which also cover the full IPv4 address space.

In the second line, we add the source IP address to be used for this route in the routing table number 101.

In the third line, we add the rule to use this routing table.

In the fourth line, we restart the interface to make the changes effective.

## Verifying the routing table

After configuring the routing table, you might want to double check that your changes to the system are correct. To do so, run the `ip` command, as shown in Example 3-4.

*Example 3-4 Double-checking your changes by using the ip command*

---

```
[root@ess5kiol ~]# ip r
default via xxxxxx dev enP1p8s0f1 proto static metric 101
[...]
10.0.12.0/24 dev enP1s0f0 proto kernel scope link src 10.0.12.102 metric 107
10.0.12.0/24 dev enP1s0f1 proto kernel scope link src 10.0.12.112 metric 108
10.0.12.0/24 dev enP51p1s0f0 proto kernel scope link src 10.0.12.103 metric 109
10.0.12.0/24 dev enP51p1s0f1 proto kernel scope link src 10.0.12.113 metric 110
[...]
192.168.12.0/24 dev enP1s0f0 proto kernel scope link src 192.168.12.102 metric 107
[...]

[root@ess5kiol ~]# ip r show table t1
0.0.0.0/1 via 10.0.12.254 dev enP1s0f0 proto static metric 107
10.0.12.0/24 dev enP1s0f0 proto static scope link src 10.0.12.102 metric 107
128.0.0.0/1 via 10.0.12.254 dev enP1s0f0 proto static metric 107
[root@ess5kiol ~]#
[root@ess5kiol ~]# ip r show table t2
0.0.0.0/1 via 10.0.12.254 dev enP1s0f1 proto static metric 108
10.0.12.0/24 dev enP1s0f1 proto static scope link src 10.0.12.112 metric 108
128.0.0.0/1 via 10.0.12.254 dev enP1s0f1 proto static metric 108
[root@ess5kiol ~]#
[root@ess5kiol ~]# ip r show table t3
0.0.0.0/1 via 10.0.12.254 dev enP51p1s0f0 proto static metric 109
10.0.12.0/24 dev enP51p1s0f0 proto static scope link src 10.0.12.103 metric 109
128.0.0.0/1 via 10.0.12.254 dev enP51p1s0f0 proto static metric 109
[root@ess5kiol ~]#
[root@ess5kiol ~]# ip r show table t4
0.0.0.0/1 via 10.0.12.254 dev enP51p1s0f1 proto static metric 110
10.0.12.0/24 dev enP51p1s0f1 proto static scope link src 10.0.12.113 metric 110
128.0.0.0/1 via 10.0.12.254 dev enP51p1s0f1 proto static metric 110
[root@ess5kiol ~]#
[root@ess5kiol ~]#
[root@ess5kiol ~]#
[root@ess5kiol ~]# ip rule show
0:      from all lookup local
32761:  from 10.0.12.102 lookup t1
32761:  from 10.0.12.112 lookup t2
32761:  from 10.0.12.103 lookup t3
32761:  from 10.0.12.113 lookup t4
32766:  from all lookup main
32767:  from all lookup default
[root@ess5kiol ~]#
```

---

### 3.4.8 The sysctl settings

You need system-wide settings for the interface scripts, which are managed by sysctl. Depending on the ESS version that is in use, you can set up the customized sysctl settings by using a *tuned profile* that is named *scale* that you set up by editing the `/etc/tuned/scale/tuned.conf` file. For any other client node in your cluster, you may deploy the same sysctl configuration file.

Use the following sysctl settings:

```
sysctl -w net.ipv6.conf.all.disable_ipv6=0
sysctl -w net.ipv6.conf.default.disable_ipv6=0
```

**Note:** IPv6 must be active.

The following example shows some recommendations for further sysctl values:

```
sysctl -w net.ipv4.conf.all.arp_ignore=2
sysctl -w net.ipv4.conf.default.arp_ignore=2
sysctl -w net.ipv4.conf.all.arp_announce=1
sysctl -w net.ipv4.conf.default.arp_announce=1
sysctl -w net.ipv4.conf.all.rp_filter=2
sysctl -w net.ipv4.conf.default.rp_filter=2
```

### 3.4.9 Applying interface settings for Differentiated Services Code Point and Priority Frame Control

The correct settings for Differentiated Services Code Point (DSCP), Type of Service (ToS), and Priority Frame Control (PFC) ensure that RoCE works without network errors. You must enable the NIC to trust DSCP. Differentiated Services (DiffServ) use a 6-bit DSCP in the 8-bit Differentiated Services field (DS field) in the IP header for packet classification purposes.

DiffServ relies on a mechanism to classify and mark packets as belonging to a specific class. DiffServ-aware routers implement per-hop behaviors (PHBs) that define the packet-forwarding properties that are associated with a class of traffic.

In addition, you must set the ToS to 106, which is the appropriate setting for (DSCP 26).

Talk to your network administrator to ensure that this configuration is supported by the network fabric. For each of the interfaces that are listed for a node by running the `ibdev2netdev` command, set their tunables.

**Note:** The settings must be applied after each restart and before IBM Spectrum Scale starts.

If you are using bonded ports, the settings must be made on the physical interface name. To do so, use the script in Example 3-5.

*Example 3-5 Script to apply interface settings for DSCP and PFC*

```
#!/bin/bash

#Set DSCP (L3) as trust mode for the NIC

for INT in `ibdev2netdev | grep Up | awk '{print $5}' | xargs`
do
```

```

if [[ $INT =~ "bond" ]] ; then
for SLAVE in `cat /proc/net/bonding/$INT | grep "Slave I" | awk '{print $3}'`
do
mlnx_qos -i $SLAVE --trust dscp
mlnx_qos -i $SLAVE --pfc 0,0,0,1,0,0,0,0
done
else
mlnx_qos -i $INT --trust dscp
mlnx_qos -i $INT --pfc 0,0,0,1,0,0,0,0
fi
done

for MLX in `ibdev2netdev | grep Up | awk '{print $1}' | xargs`
do
echo 106 > /sys/class/infiniband/${MLX}/tc/1/traffic_class
cma_roce_tos -d $MLX -t 106
done

```

---

Run this script after each restart of network-scripts. The `mlnx_qos` command does not work on virtual devices, which includes bonding.

## PCI slot settings

On older existing IBM POWER8® models, there are reasonable performance improvements by setting the `INT_LOG_MAX_PAYLOAD_SIZE` parameter to 4k. There also was better performance observed on other architectures like AMD64, IBM POWER9™, and Intel machines.

There were cases that we tested where we did not achieve better performance numbers by setting this parameter, but we never saw a negative impact. Therefore, you should set this parameter.

Changing this setting requires a restart of the server to become effective. To adjust the parameter, see Example 3-6.

### *Example 3-6 Adjusting the INT\_LOG\_MAX\_PAYLOAD parameter*

```

# ls /sys/class/infiniband/
mlx5_0  mlx5_1  mlx5_2  mlx5_3  mlx5_bond_0

# set it to 4k
for i in `ls /sys/class/infiniband/`; do mlxconfig -y -d $i -e s
INT_LOG_MAX_PAYLOAD_SIZE=12; done
#check
for i in `ls /sys/class/infiniband/`; do mlxconfig -d $i -e q
INT_LOG_MAX_PAYLOAD_SIZE; done
#

```

---





## Functional verification of the network

The configuration should be tested on different layers. To verify that the interface and network settings are correct, use the **nsdperf** tool, **ping**, and the **ib\_\*tools** set from Mellanox OpenFabrics Enterprise Distribution (MOFED). To obtain a baseline for comparison, you should run a first cycle with TCP/IP only.

## 4.1 Quick test: ping

Try to ping from each interface to the other interface between two nodes, as shown in Example 4-1. Make sure that you include each local interface and remote IP address and loop over all of them. Also, ping with a larger packet size to check that the RDMA over Converged Ethernet (RoCE) packets with larger sizes can be transferred in the fabric.

*Example 4-1 Pinging from each interface to the other interface between two nodes*

---

```
#local
[root@ess5kio1]# ip -4 a | grep 10.0.12 | grep inet
    inet 10.0.12.102/24 brd 10.0.12.255 scope global noprefixroute enp1s0f0
    inet 10.0.12.112/24 brd 10.0.12.255 scope global noprefixroute enp1s0f1
    inet 10.0.12.103/24 brd 10.0.12.255 scope global noprefixroute enP51pls0f0
    inet 10.0.12.113/24 brd 10.0.12.255 scope global noprefixroute enP51pls0f1
[root@ess5kio1]#

# remote
[root@ess5kio2 ~]# ip -4 a | grep 10.0.12 | grep inet
    inet 10.0.12.104/24 brd 10.0.12.255 scope global noprefixroute enp1s0f0
    inet 10.0.12.114/24 brd 10.0.12.255 scope global noprefixroute enp1s0f1
    inet 10.0.12.105/24 brd 10.0.12.255 scope global noprefixroute enP51pls0f0
    inet 10.0.12.115/24 brd 10.0.12.255 scope global noprefixroute enP51pls0f1
[root@ess5kio2 ~]#

## loop
[root@ess5kio1 profile.d]# for LocalIP in `ip -4 a | grep 10.0.12 | grep inet | awk '{print $2}'
| cut -d / -f 1`
> do
> for RemoteIP in `ssh root@ess5kio2 "ip -4 a | grep 10.0.12 | grep inet " | awk '{print $2}' |
cut -d / -f 1`
> do
> ping $RemoteIP -I $LocalIP -c 1 -s 4096 | grep "4104 bytes from";
> done
> done

4104 bytes from 10.0.12.104: icmp_seq=1 ttl=64 time=0.060 ms
4104 bytes from 10.0.12.114: icmp_seq=1 ttl=64 time=0.062 ms
4104 bytes from 10.0.12.105: icmp_seq=1 ttl=64 time=0.073 ms
4104 bytes from 10.0.12.115: icmp_seq=1 ttl=64 time=0.063 ms
4104 bytes from 10.0.12.104: icmp_seq=1 ttl=64 time=0.050 ms
4104 bytes from 10.0.12.114: icmp_seq=1 ttl=64 time=0.047 ms
4104 bytes from 10.0.12.105: icmp_seq=1 ttl=64 time=0.059 ms
4104 bytes from 10.0.12.115: icmp_seq=1 ttl=64 time=0.065 ms
4104 bytes from 10.0.12.104: icmp_seq=1 ttl=64 time=0.059 ms
4104 bytes from 10.0.12.114: icmp_seq=1 ttl=64 time=0.062 ms
4104 bytes from 10.0.12.105: icmp_seq=1 ttl=64 time=0.081 ms
4104 bytes from 10.0.12.115: icmp_seq=1 ttl=64 time=0.065 ms
4104 bytes from 10.0.12.104: icmp_seq=1 ttl=64 time=0.048 ms
4104 bytes from 10.0.12.114: icmp_seq=1 ttl=64 time=0.054 ms
4104 bytes from 10.0.12.105: icmp_seq=1 ttl=64 time=0.085 ms
4104 bytes from 10.0.12.115: icmp_seq=1 ttl=64 time=0.056 ms
[root@ess5kio1 profile.d]#
```

---

## 4.2 Quick test: ib\_read\_bw

Use the **ib\_read\_bw** tool with the **-R** parameter for using the Connection Manager (CM). The **ib\_[read,write]\_[bw,lat]** toolset from MOFED can be used on only one interface at a time. However, it is an essential toolset for validating that the network is healthy. On one (server-) node, open a session and start the tool on that server mode, as shown in Example 4-2.

Example 4-2 The **ib\_read\_bw -R** toolset

```
[root@ess5kio1 profile.d]# ip -4 a | grep 10.0.12 | grep inet | awk '{print $2}' | cut -d /  
-f 1  
10.0.12.102  
10.0.12.112  
10.0.12.103  
10.0.12.113  
[root@ess5kio1 profile.d]# ib_read_bw -R
```

```
*****  
* Waiting for client to connect... *  
*****
```

Open a second terminal on the other node and connect to one of the IP addresses that are shown in Example 4-2. Make sure that no **down** verbs interfaces are available. If the machine has unused ports, specify the active port, as shown in Example 4-3. Repeat this step for each target connection.

Example 4-3 **RDMA\_Read BW Test**

```
[root@ess5kio2 ~]# ip -4 a | grep 10.0.12 | grep inet  
inet 10.0.12.104/24 brd 10.0.12.255 scope global noprefixroute enp1s0f0  
inet 10.0.12.114/24 brd 10.0.12.255 scope global noprefixroute enp1s0f1  
inet 10.0.12.105/24 brd 10.0.12.255 scope global noprefixroute enP51p1s0f0  
inet 10.0.12.115/24 brd 10.0.12.255 scope global noprefixroute enP51p1s0f1  
[root@ess5kio2 ~]# ib_read_bw -R 10.0.12.102
```

```
-----  
RDMA_Read BW Test  
Dual-port      : OFF      Device      : mlx5_1  
Number of qps  : 1        Transport type : IB  
Connection type : RC      Using SRQ      : OFF  
PCIe relax order: ON  
TX depth      : 128  
CQ Moderation  : 1  
Mtu           : 4096[B]  
Link type      : Ethernet  
GID index      : 4  
Outstand reads : 16  
rdma_cm QPs    : ON  
Data ex. method : rdma_cm  
-----  
local address: LID 0000 QPN 0x013b PSN 0xf99618  
GID: 00:00:00:00:00:00:00:00:00:00:255:255:10:00:12:104  
remote address: LID 0000 QPN 0x0122 PSN 0x278238  
GID: 00:00:00:00:00:00:00:00:00:00:255:255:10:00:12:102  
-----  
#bytes    #iterations    BW peak[MB/sec]    BW average[MB/sec]    MsgRate[Mpps]  
65536     1000             11675.02           11673.95              0.186783
```

---

```
[root@ess5kio2 ~]#
```

---

If you see results that are shown in Example 4-3 on page 25, then your network works as expected. The bandwidth of 1,167,395 MBps is a good value for a 100 GbE network. A value below 11 GBps indicates potential problems with the network. You can now redo the tests with other options and adapter combinations, as shown in Example 4-4. Start as before on node 1.

*Example 4-4 Command-line output for `ib_read_test`*

---

```
[root@ess5kio1 profile.d]# ibdev2netdev
mlx5_0 port 1 ==> enp1s0f0 (Up)
mlx5_1 port 1 ==> enp1s0f1 (Up)
mlx5_2 port 1 ==> enP51p1s0f0 (Up)
mlx5_3 port 1 ==> enP51p1s0f1 (Up)
[root@ess5kio1 profile.d]# ip -4 a | grep inet | grep enP51p1s0f0
    inet 10.0.12.103/24 brd 10.0.12.255 scope global noprefixroute enP51p1s0f0
[root@ess5kio1 profile.d]# ib_read_bw -R --ib-dev=mlx5_2

*****
* Waiting for client to connect... *
*****
```

---

On the second node, pick one of the available RoCE adapters and run the `ib_*` command, as shown by `ib_read_test (2/2)` in Example 4-5.

*Example 4-5 Command line for `ib_read_test (2/2)`*

---

```
[root@ess5kio2 ~]# ibdev2netdev
mlx5_0 port 1 ==> enp1s0f0 (Up)
mlx5_1 port 1 ==> enp1s0f1 (Up)
mlx5_2 port 1 ==> enP51p1s0f0 (Up)
mlx5_3 port 1 ==> enP51p1s0f1 (Up)
[root@ess5kio2 ~]# ib_read_bw --ib-dev=mlx5_3 -R 10.0.12.103
[...]
```

---

#bytes	#iterations	BW peak[MB/sec]	BW average[MB/sec]	MsgRate[Mpps]
65536	1000	11679.30	11675.39	0.186806

---

```
[root@ess5kio2 ~]#
```

---

## 4.3 Nsdperf test with TCP

You can use the `nsdperf` tool to measure the network connection speed between multiple nodes all at once. The connection itself and the bandwidth capabilities of the network are checked.

Multiple enhancements are in `nsdperf`. The `nsdperf` tool has a version function, and you can check the version (the source is in `/usr/lpp/mmfs/samples/net`) by running the command that is shown in Example 4-6 on page 27.

---

*Example 4-6 Checking the nsdperf version*

---

```
[root@fsgcc-sr650-13 net]# ./nsdperf -s -d
nsdperf 1.29 server started
^C
[root@fsgcc-sr650-13 net]#
```

---

For multiple adapters to be used for RoCE, you need to be using Version 1.29 at a minimum. Create a command file that includes all the IBM Elastic Storage System (ESS) server nodes and clients, as shown in Example 4-7.

---

*Example 4-7 Create a command file*

---

```
[root@ess5-ems1 nsdperf]# cat nsdperf_commandfile
s ess5kio1
s ess5kio2
c ece-13.mmfsd.net
c ece-15.mmfsd.net
c ece-14.mmfsd.net
c ece-16.mmfsd.net
c ece-17.mmfsd.net
c ece-18.mmfsd.net
c ece-19.mmfsd.net
c ece-20.mmfsd.net
ttime 30
threads 12
buffsize 8388608
hist off
status
t w
t r
reset
quit
```

---

Start **nsdperf** in server mode on all nodes by running the following command:

```
mmdsh -N all "/usr/lpp/mmfs/samples/net/nsdperf -s"
```

Run the **nsdperf** command on the node, where the command file is in the file that is shown in our Example 4-7. The **bold** lines in Example 4-8 show the results. As you can see, multiple clients connecting to the two servers per Internet Protocol network produces bandwidth of ~ 24 GBps, which is nearly the line speed of the network.

---

*Example 4-8 The nsdperf command results*

---

```
[root@ess5-ems1 nsdperf]# /usr/lpp/mmfs/samples/net/nsdperf -i nsdperf_commandfile
Connected to ess5kio1
Connected to ess5kio2
Connected to ece-13.mmfsd.net
Connected to ece-15.mmfsd.net
Connected to ece-14.mmfsd.net
Connected to ece-16.mmfsd.net
Connected to ece-17.mmfsd.net
Connected to ece-18.mmfsd.net
Connected to ece-19.mmfsd.net
Connected to ece-20.mmfsd.net
Test time set to 30 seconds
```

```
Number of tester threads set to 12
Buffer size set to 8388608 bytes
Histogram printing is now off
test time: 30 sec
data buffer size: 8388608
TCP socket send/receive buffer size: 0
tester threads: 12
parallel connections: 1
RDMA enabled: no
```

clients:

```
ece-13.mmfsd.net (192.168.12.13)
ece-14.mmfsd.net (192.168.12.14)
ece-15.mmfsd.net (192.168.12.15)
ece-16.mmfsd.net (192.168.12.16)
ece-17.mmfsd.net (192.168.12.17)
ece-18.mmfsd.net (192.168.12.18)
ece-19.mmfsd.net (192.168.12.119)
ece-20.mmfsd.net (192.168.12.210)
```

servers:

```
ess5kio1 (192.168.12.102)
ess5kio2 (192.168.12.104)
```

**8-2 write 24000 MB/sec (2860 msg/sec), cli 4% srv 11%, time 30, buff 8388608, th 12**

**8-2 read 24700 MB/sec (2950 msg/sec), cli 5% srv 10%, time 30, buff 8388608, th 12**

[root@ess5-ems1 nsdperf]#

---

As you can see in Example 4-8 on page 27, the network bandwidth is what you would expect from a POWER8 processor-based ESS when you run with TCP traffic and use one interface per machine.

## 4.4 The nsdperf command with Remote Direct Memory Access

Use or create a command file that contains the two entries that are highlighted in **bold** in Example 4-9. The **rdma** (Remote Direct Memory Access (RDMA)) and **usecm** parameters enable **nsdperf** to use RoCE.

*Example 4-9 Run nsdperf with the created command file*

---

```
[root@ess5-ems1 nsdperf]# cat nsdperf_commandfile
s ess5kio1
s ess5kio2
c ece-13.mmfsd.net
c ece-15.mmfsd.net
c ece-14.mmfsd.net
c ece-16.mmfsd.net
c ece-17.mmfsd.net
c ece-18.mmfsd.net
c ece-19.mmfsd.net
c ece-20.mmfsd.net
ttime 30
threads 12
buffsize 8388608
hist off
status
```

```

rdma on
usecm on
t w
t r
reset
quit
[root@ess5-ems1 nsdperf]#

[root@ess5-ems1 nsdperf]# mmdsh -N all "/usr/lpp/mmfs/samples/net/nsdperf -r
mlx5_0/1,mlx5_1/2,mlx5_2/1,mlx5_3/1 -s "

```

---

Start **nsdperf** in server mode on all machines, as shown in Example 4-10.

*Example 4-10 Starting nsdperf in server mode on all machines*

---

```

# Storage cluster
[root@ess5-ems1 nsdperf]# mmdsh -N all "kill nsdperf"
[root@ess5-ems1 nsdperf]# mmdsh -N all "/usr/lpp/mmfs/samples/net/nsdperf -r \
mlx5_0/1,mlx5_1/2,mlx5_2/1,mlx5_3/1 -s "

# client cluster
[root@fscs-sr650-13 hadoop2]# mmdsh -N all "kill nsdperf"
[root@fscs-sr650-13 hadoop2]# mmdsh -N all "/usr/lpp/mmfs/samples/net/nsdperf -s -r mlx5_0/1"

[root@ess5-ems1 nsdperf]# /usr/lpp/mmfs/samples/net/nsdperf -i nsdperf_commandfile
Connected to ess5kio1
Connected to ess5kio2
Connected to ece-13.mmfsd.net
Connected to ece-15.mmfsd.net
Connected to ece-14.mmfsd.net
Connected to ece-16.mmfsd.net
Connected to ece-17.mmfsd.net
Connected to ece-18.mmfsd.net
Connected to ece-19.mmfsd.net
Connected to ece-20.mmfsd.net
Test time set to 30 seconds
Number of tester threads set to 12
Buffer size set to 8388608 bytes
Histogram printing is now off
test time: 30 sec
data buffer size: 8388608
TCP socket send/receive buffer size: 0
tester threads: 12
parallel connections: 1
RDMA enabled: no

clients:
ece-13.mmfsd.net (192.168.12.13)
  mlx5_0:1 192.168.12.13/255.255.255.0 10.0.12.13/255.255.255.0 ba59:9fff:fe27:a404
ece-14.mmfsd.net (192.168.12.14)
  mlx5_0:1 192.168.12.14/255.255.255.0 10.0.12.14/255.255.255.0 ba59:9fff:fe27:a488
ece-15.mmfsd.net (192.168.12.15)
  mlx5_0:1 192.168.12.15/255.255.255.0 10.0.12.15/255.255.255.0 ba59:9fff:fe27:a580
ece-16.mmfsd.net (192.168.12.16)
  mlx5_0:1 192.168.12.16/255.255.255.0 10.0.12.16/255.255.255.0 ba59:9fff:fe27:a470
ece-17.mmfsd.net (192.168.12.17)

```

```

mlx5_0:1 192.168.12.17/255.255.255.0 10.0.12.17/255.255.255.0 ba59:9fff:fe27:a400
ece-18.mmfsd.net (192.168.12.18)
mlx5_0:1 192.168.12.18/255.255.255.0 10.0.12.18/255.255.255.0 ba59:9fff:fe4c:6fdc
ece-19.mmfsd.net (192.168.12.119)
mlx5_0:1 192.168.12.119/255.255.255.0 10.0.12.119/255.255.255.0 ba59:9fff:fe27:a4d8
ece-20.mmfsd.net (192.168.12.210)
mlx5_0:1 192.168.12.210/255.255.255.0 10.0.12.210/255.255.255.0 ba59:9fff:fe4c:6f80
servers:
ess5kio1 (192.168.12.102)
mlx5_0:1 10.0.12.102/255.255.255.0 192.168.12.102/255.255.255.0 0e42:a1ff:fed7:1ae8
mlx5_2:1 10.0.12.103/255.255.255.0 0e42:a1ff:fed7:1bdc
mlx5_3:1 10.0.12.113/255.255.255.0 0e42:a1ff:fed7:1bdd
ess5kio2 (192.168.12.104)
mlx5_0:1 10.0.12.104/255.255.255.0 192.168.12.104/255.255.255.0 0e42:a1ff:feed:c108
mlx5_2:1 10.0.12.105/255.255.255.0 0e42:a1ff:feed:c104
mlx5_3:1 10.0.12.115/255.255.255.0 0e42:a1ff:feed:c105
RDMA is now on
Connection Manager is now enabled
8-2 write 72100 MB/sec (8600 msg/sec), cli 3% srv 1%, time 30, buff 8388608, th 12, RDMA
8-2 read 73500 MB/sec (8760 msg/sec), cli 2% srv 5%, time 30, buff 8388608, th 12, RDMA
[root@ess5-ems1 nsdperf]#

```

---

The results depend on the model that you use. The recent ESS 5000 models can run network traffic to saturate nearly four 100 GbE links. Future models might come with even higher bandwidth to operate efficiently in a 200 High Dynamic Range (HDR) network.

After the network shows acceptable results, you can proceed and configure IBM Spectrum Scale.

## 4.5 Debugging method for a network by using system on-board tools

This section describes what you can do if you do not see the expected bandwidth and you want to generate more measurements for further debugging. As an example, we have a node with a dual-port 100 Enhanced Data Rate (EDR) GbE connection and another node with four ports. The OpenFabrics Enterprise Distribution (OFED) standard tools cannot be used as is for running a workload with multiple ports, but we can run multiple instances to accomplish this task. If we use more than one adapter, understand that OFED tools like **ib\_read\_bw** and **ib\_write\_bw** are missing the feature to specify a specific adapter port to use for the test when initiating the **ib\_[read,write,send]\_bw** command. Therefore, the easiest way to accomplish our goal is to temporarily add IP aliases to the adapters. By using these temporary IP addresses in the test, a clear relationship to the port is ensured.

Our example has EMS NodeA (Example 4-11) and Server NodeB (Example 4-12 on page 31).

### *Example 4-11 EMS NodeA*

```

[root@ems5000 ~]# ibdev2netdev
mlx5_0 port 1 ==> enp1s0f0 (Up)
mlx5_1 port 1 ==> enp1s0f1 (Up)
[root@ems5000 ~]#
[root@ems5000 ~]# ip a add 192.168.50.100/24 dev enp1s0f0

```

```
[root@ems5000 ~]# ip a add 192.168.51.100/24 dev enp1s0f1
[root@ems5000 ~]# ip -4 a | grep inet | grep 192
    inet 192.168.50.100/24 scope global enp1s0f0
    inet 192.168.51.100/24 scope global enp1s0f1
```

#### Example 4-12 Server NodeB

```
[root@ess5kiol ~]# ibdev2netdev
mlx5_0 port 1 ==> enp1s0f0 (Up)
mlx5_1 port 1 ==> enp1s0f1 (Up)
[...]
[root@ess5kiol ~]# ip a add 192.168.50.101/24 dev enp1s0f0
[root@ess5kiol ~]# ip a add 192.168.51.101/24 dev enp1s0f1
[root@ess5kiol ~]#
```

Open a session on each adapter of the “server” node but with a different port:

```
SCREEN 1: [root@ess5kiol ~]# ib_read_bw -d mlx5_0 -R --run_infinately
SCREEN 2: [root@ess5kiol ~]# ib_read_bw -d mlx5_1 -R -p 10000 --run_infinately
```

Open two sessions on Node A:

```
[root@ems5000 ~]# ib_read_bw -d mlx5_0 -R 192.168.50.101 --run_infinately
[root@ems5000 ~]# ib_read_bw -d mlx5_1 -R 192.168.51.101 -p 10000 --run_infinately
```

Run the test so that your window looks like the example that is shown in Figure 4-1.

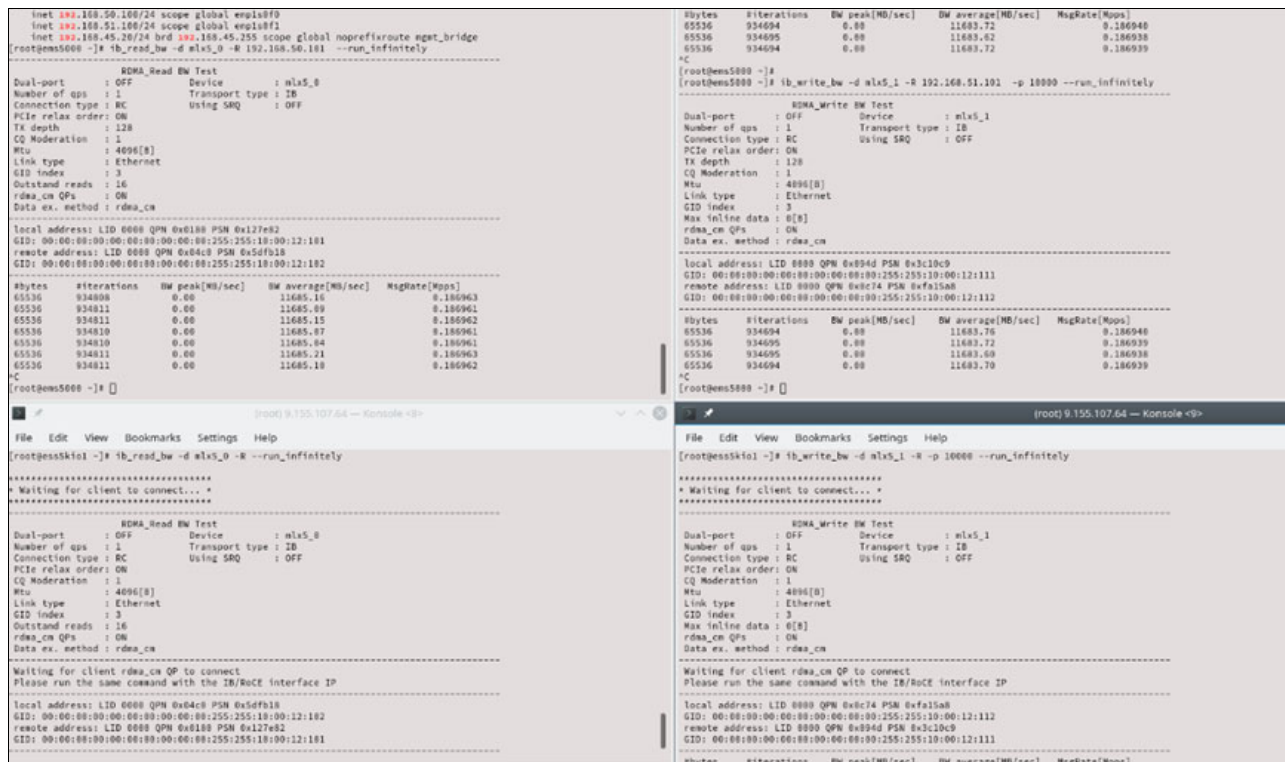


Figure 4-1 Command-line example: Parallel OFED tool test

You should get output from both connections simultaneously. Remember to remove the temporary IP address from the adapters after you are done.





## IBM Spectrum Scale settings for Remote Direct Memory Access over Converged Ethernet

After the network interfaces and the network are configured to run RDMA over Converged Ethernet (RoCE), you must adjust the IBM Spectrum Scale settings to use Remote Direct Memory Access (RDMA). Before proceeding with the IBM Spectrum Scale configuration, as a best practice, verify that the fabric configuration works with the current interface settings. For more information about testing the configuration, see Chapter 4, “Functional verification of the network” on page 23.

To enable RoCE in IBM Spectrum Scale, run the **mmchconfig** command to set **verbsRdma=enable** and **verbsRdmaCm=enable**, as shown in Example 5-1.

*Example 5-1 Running the mmchconfig command for the verbsRDMA settings*

---

```
[root@ess5-ems1 nsdperf]# mmclsconfig | grep verbsRdma
verbsRdma enable
verbsRdmaSend yes
verbsRdmaCm enable
[root@ess5-ems1 nsdperf]#
```

---

Add the ports that you need for your node's adapter configuration, as shown in Example 5-2.

*Example 5-2 Ports for the node's adapter configuration*

---

```
[root@ess5-ems1 nsdperf]# mmclsconfig | grep verbsPort

# one common fabric
verbsPorts mlx5_0/1 mlx5_1/1 mlx5_2/1 mlx5_3/1

# fabric 3 + 4
verbsPorts mlx5_0/1/3 mlx5_2/1/3 mlx5_1/1/4 mlx5_3/1/4

[root@ess5-ems1 nsdperf]#
```

---

I After applying the changes, restart IBM Spectrum Scale and verify that the connections are in place by running **mmdiag --network**.

In IBM Spectrum Scale V5.0.4 and later, the IBM Spectrum Scale daemon startup service waits for a specified period for the RDMA ports on a node to become active. You can adjust the length of the timeout period and choose the action that the startup service takes if the timeout period expires.

For more information, see the descriptions of the **verbsPortsWaitTimeout** attribute and the **verbsRdmaFailBackTCPIfNotAvailable** attribute in the **mmchconfig** [command topic](#).

To verify that RoCE and IBM Spectrum Scale are working properly, see Chapter 6, “Functional verification of IBM Spectrum Scale” on page 35.

# Functional verification of IBM Spectrum Scale

After starting IBM Spectrum Scale, you will see that IBM Spectrum Scale is using **verbsPorts**. You might need to access the file system on the clients first to force the node to connect to the IBM Elastic Storage System (ESS). To check that there is at least one active queue pair (QP) per adapter, run the **mmdiag -network** command, as shown in Example 6-1.

**Example 6-1** Remote Direct Memory Access connections between nodes

```
RDMA Connections between nodes:
Fabric 3 - Device mlx5_0 Port 1 Width 4x Speed EDR lid 0
  hostname      idx CM state VS buff RDMA_CT(ERR) RDMA_RCV_MB RDMA_SND_MB VS_CT(ERR) VS_SND_MB VS_RCV_MB
WAIT_CON_SLOT WAIT_NODE_SLOT
ems5k.mmfsd      0  Y  RTS  (Y)256 5006 (0 ) 1007      289      130173(0 ) 13      11      0
0
ess5kio2.mmfsd   0  Y  RTS  (Y)256 435755 (0 ) 3408      1196      2665404(0 ) 6246      253      0
0
ece-14.mmfsd     0  Y  RTS  (Y)256 65 (0 ) 0      16      3604 (0 ) 0      0      0
0
ece-15.mmfsd     0  Y  RTS  (Y)256 65 (0 ) 0      16      3648 (0 ) 0      0      0
0
ece-17.mmfsd     0  Y  RTS  (Y)256 65 (0 ) 0      16      3602 (0 ) 0      0      0
0
ece-13.mmfsd     0  Y  RTS  (Y)256 61345 (0 ) 509      30      8046592(0 ) 2294      3688      0
0
ece-16.mmfsd     0  Y  RTS  (Y)256 128 (0 ) 0      33      3602 (0 ) 0      0      0
0
ece-18.mmfsd     0  Y  RTS  (Y)256 639 (0 ) 117      48      2780209(0 ) 841      1024      0
0
Fabric 3 - Device mlx5_2 Port 1 Width 4x Speed EDR lid 0
  hostname      idx CM state VS buff RDMA_CT(ERR) RDMA_RCV_MB RDMA_SND_MB VS_CT(ERR) VS_SND_MB VS_RCV_MB
WAIT_CON_SLOT WAIT_NODE_SLOT
ece-17.mmfsd     1  Y  RTS  (Y)256 128 (0 ) 0      33      3602 (0 ) 0      0      0
0
ece-15.mmfsd     1  Y  RTS  (Y)256 128 (0 ) 0      33      3648 (0 ) 0      0      0
0
ece-13.mmfsd     1  Y  RTS  (Y)256 61531 (0 ) 499      69      8045744(0 ) 2291      3682      0
0
ece-18.mmfsd     1  Y  RTS  (Y)256 684 (0 ) 127      50      2780190(0 ) 842      1024      0
0
ece-14.mmfsd     1  Y  RTS  (Y)256 128 (0 ) 0      33      3603 (0 ) 0      0      0
0
ems5k.mmfsd      1  Y  RTS  (Y)256 4999 (0 ) 941      354      130171(0 ) 13      11      0
0
ece-16.mmfsd     1  Y  RTS  (Y)256 64 (0 ) 0      16      3602 (0 ) 0      0      0
0
Fabric 4 - Device mlx5_1 Port 1 Width 4x Speed EDR lid 0
```

hostname	idx	CM	state	VS	buff	RDMA_CT(ERR)	RDMA_RCV_MB	RDMA_SND_MB	VS_CT(ERR)	VS_SND_MB	VS_RCV_MB
Fabric 4 - Device mlx5_3 Port 1 Width 4x Speed EDR lid 0											
WAIT_CON_SLOT WAIT_NODE_SLOT	idx	CM	state	VS	buff	RDMA_CT(ERR)	RDMA_RCV_MB	RDMA_SND_MB	VS_CT(ERR)	VS_SND_MB	VS_RCV_MB
ece-20.mmfsd	0	Y	RTS	(Y)256	695	(0 ) 134	44	2698124(0 )	810	988	0
0											
ess5kio2.mmfsd	1	Y	RTS	(Y)256	435882	(0 ) 3414	1196	2665403(0 )	6247	254	0
0											
ece-19.mmfsd	0	Y	RTS	(Y)256	503	(0 ) 58	71	2851219(0 )	855	1041	0
0											
Fabric 3 - Device mlx5_0 Port 1 Width 4x Speed EDR lid 0											
WAIT_CON_SLOT WAIT_NODE_SLOT	idx	CM	state	VS	buff	RDMA_CT(ERR)	RDMA_RCV_MB	RDMA_SND_MB	VS_CT(ERR)	VS_SND_MB	VS_RCV_MB
ess5kio2.mmfsd	2	Y	RTS	(Y)256	435862	(0 ) 3410	1232	2665400(0 )	6246	254	0
0											
ece-19.mmfsd	1	Y	RTS	(Y)256	466	(0 ) 88	32	2851194(0 )	858	1045	0
0											
ece-20.mmfsd	1	Y	RTS	(Y)256	732	(0 ) 134	54	2698131(0 )	811	985	0
0											

In our example, a client node has four QPs because it connects to the ESS I/O nodes with two connections each, as shown in Example 6-2.

**Example 6-2** Client node with four QPs that are connected to the ESS IO nodes with two connections each

RDMA Connections between nodes:											
Fabric 3 - Device mlx5_0 Port 1 Width 4x Speed EDR lid 0											
hostname	idx	CM	state	VS	buff	RDMA_CT(ERR)	RDMA_RCV_MB	RDMA_SND_MB	VS_CT(ERR)	VS_SND_MB	VS_RCV_MB
Fabric 3 - Device mlx5_0 Port 1 Width 4x Speed EDR lid 0											
WAIT_CON_SLOT WAIT_NODE_SLOT	idx	CM	state	VS	buff	RDMA_CT(ERR)	RDMA_RCV_MB	RDMA_SND_MB	VS_CT(ERR)	VS_SND_MB	VS_RCV_MB
ece-15.mmfsd	0	Y	RTS	(Y)256	372560	(0 ) 38	19171	273596(0 )	22	32	0
0											
[...]											
ess5kio1.mmfsd	0	Y	RTS	(Y)256	0	(0 ) 0	0	8046592(0 )	3688	2294	0
0											
ess5kio1.mmfsd	1	Y	RTS	(Y)256	0	(0 ) 0	0	8045744(0 )	3682	2291	0
0											
ess5kio2.mmfsd	0	Y	RTS	(Y)256	0	(0 ) 0	0	375556(0 )	926	546	0
0											
ess5kio2.mmfsd	1	Y	RTS	(Y)256	0	(0 ) 0	0	375554(0 )	925	545	0
0											
[...]											
ece-14.mmfsd	0	Y	RTS	(Y)256	0	(0 ) 0	0	4 (0 ) 0	0	0	0
0											
[root@fscs-sr650-13 ~]#											



# Mellanox switch configuration for Remote Direct Memory Access over Converged Ethernet

If you are using a Mellanox switch, it is simple to configure the network for RDMA over Converged Ethernet (RoCE). Example A-1 shows a configuration example for running the Mellanox Onyx operating system (OS). You may also use CUMULUS.

Check the status and set it as shown in Example A-1.

## *Example A-1 Mellanox Onyx operating system configuration*

---

```
[root@ess5-ems1 ~]# ssh sw1 -l admin
Mellanox Onyx Switch Management
Password:
Last login: Mon May 28 17:51:05 UTC 2001 from 9.152.186.100 on pts/0
Number of total successful connections since last 30 days: 5
```

Mellanox Switch

```
mestore-sw100-3 [ess5k-sharky: master] > en
mestore-sw100-3 [ess5k-sharky: master] # conf t

mestore-sw100-3 [ess5k-sharky: master] (config) # roce lossless
mestore-sw100-3 [ess5k-sharky: master] (config) # show roce
```

```
RoCE mode      : lossless
LLDP           : enabled
Port trust mode: L3
```

Application TLV:

```
Selector: udp
Protocol: 4791
Priority: 3
```

Port congestion-control:

Mode: ecn, absolute  
Min : 150  
Max : 1500

PFC : enabled  
switch-priority 3: enabled

RoCE used TCs:

Switch-Priority	TC	Application	ETS
3	3	RoCE	WRR 50%
6	6	CNP	Strict

RoCE buffer pools:

Traffic Pool	Type	Memory [%]	Switch Priorities	Memory actual	Usage	Max Usage
lossy-default	lossy	auto	0, 1, 2, 4, 5, 6, 7	5.4M	0	5.3M
roce-reserved	lossless	auto	3	5.4M	0	4.8M

Exception list:  
N/A

mestore-sw100-3 [ess5k-sharky: master] (config) #

For this example, you log in to the switches by using admin as the username and password.

# Configuring Multi-Chassis Link Aggregation on Mellanox switches

For more information about how to configure Multi-Chassis Link Aggregation (MLAG) on Mellanox switches, see [How to Configure MLAG on Mellanox Switches](#). Follow the documentation to initially configure Inter-switch Link (ISL) communication. After initially configuring the switches with ISLs, you can use a configuration like the one that is shown in Example B-1.

## *Example B-1 Switch configuration steps*

---

```
mestore-sw100-3 [ess5k-sharky: master] (config) # interface mlag-port-channel 1 description io1-enpl-bond0
mestore-sw100-3 [ess5k-sharky: master] (config) # interface mlag-port-channel 2 description io2-enpl-bond0
mestore-sw100-3 [ess5k-sharky: master] (config) #

mestore-sw100-4 [ess5k-sharky: standby] (config) # interface mlag-port-channel 1 description io1-enpl-bond0
mestore-sw100-4 [ess5k-sharky: standby] (config) # interface mlag-port-channel 2 description io2-enpl-bond0
mestore-sw100-4 [ess5k-sharky: standby] (config) #

## set MTU
mestore-sw100-3 [ess5k-sharky: master] (config) # interface mlag-port-channel 1-2 mtu 9000 force
mestore-sw100-4 [ess5k-sharky: standby] (config) # interface mlag-port-channel 1-2 mtu 9000 force

## now sort the ethernet ports to the channel

mestore-sw100-3 [ess5k-sharky: master] (config) # interface ethernet 1/12 speed 100G force
mestore-sw100-3 [ess5k-sharky: master] (config) # interface ethernet 1/12 mlag-channel-group 1 mode active

otherswitch:
mestore-sw100-4 [ess5k-sharky: standby] (config) # interface ethernet 1/6 mlag-channel-group 1 mode active
mestore-sw100-4 [ess5k-sharky: standby] (config) # interface mlag-port-channel 1-2 mtu 9000 force

# start on both sides
no interface mlag-port-channel 1 shutdown
no interface mlag-port-channel 1 shutdown
```

---



# Related publications

The publications that are listed in this section are considered suitable for a more detailed description of the topics that are covered in this paper.

## IBM Redbooks

The following IBM Redbooks publications provide more information about the topics in this document. Some publications that are referenced in this list might be available in softcopy only.

- ▶ *IBM Spectrum Scale and IBM Elastic Storage System Network Guide*, REDP-5484
- ▶ *Implementation Guide for IBM Elastic Storage System 3000*, SG24-8443
- ▶ *Implementation Guide for IBM Elastic Storage System 3200*, SG24-8516
- ▶ *Implementation Guide for IBM Elastic Storage System 5000*, SG24-8498
- ▶ *Introduction Guide to the IBM Elastic Storage System*, REDP-5253

You can search for, view, download, or order these documents and other Redbooks, Redpapers, web docs, drafts, and additional materials at the following website:

[ibm.com/redbooks](https://ibm.com/redbooks)

## Online resources

These websites are also relevant as further information sources:

- ▶ Configuring network bonding  
[https://access.redhat.com/documentation/en-us/red\\_hat\\_enterprise\\_linux/8/html/configuring\\_and\\_managing\\_networking/configuring-network-bonding\\_configuring-and-managing-networking](https://access.redhat.com/documentation/en-us/red_hat_enterprise_linux/8/html/configuring_and_managing_networking/configuring-network-bonding_configuring-and-managing-networking)
- ▶ How To Configure MLAG on Mellanox Switches  
<https://community.mellanox.com/s/article/how-to-configure-mlag-on-mellanox-switches>
- ▶ IBM Elastic Storage Server Quick Deployment Guide  
<https://www.ibm.com/docs/en/ess-p8/6.1.0?topic=quick-deployment-guide>
- ▶ IBM Spectrum Scale `mmchconfig` command  
<https://www.ibm.com/docs/en/spectrum-scale/5.1.1?topic=reference-mmchconfig-command>
- ▶ IBM Spectrum Scale Overview  
<https://www.ibm.com/docs/en/STXKQY/gpfscclustersfaq.html#fsm>
- ▶ Mellanox: Recommended Network Configuration Examples for RoCE Deployment  
<https://community.mellanox.com/s/article/recommended-network-configuration-examples-for-roce-deployment>

## Help from IBM

IBM Support and downloads

[ibm.com/support](https://ibm.com/support)

IBM Global Services

[ibm.com/services](https://ibm.com/services)

# Abbreviations and acronyms

<b>ATS</b>	Advanced Technical Support
<b>CM</b>	Connection Manager
<b>DiffServ</b>	Differentiated Services
<b>DSCP</b>	Differentiated Services Code Point
<b>EDR</b>	Enhanced Data Rate
<b>ESCC</b>	European Storage Competence Center
<b>ESS</b>	IBM Elastic Storage System
<b>HA</b>	high availability or highly available
<b>HDR</b>	High Dynamic Range
<b>HPC</b>	high-performance computing
<b>IBM</b>	International Business Machines Corporation
<b>ISL</b>	Inter-switch Link
<b>LACP</b>	Link Aggregation Control Protocol
<b>MLAG</b>	Multi-Chassis LAG Multi-Chassis Link Aggregation
<b>MOFED</b>	Mellanox OpenFabrics Enterprise Distribution
<b>MTU</b>	Maximum Transmission Unit
<b>NIC</b>	Network Interface Controller
<b>NSD</b>	Network Shared Disk
<b>OFED</b>	OpenFabrics Enterprise Distribution
<b>OS</b>	operating system
<b>PFC</b>	Priority Frame Control
<b>PHB</b>	per-hop behavior
<b>QP</b>	queue pair
<b>RDMA</b>	Remote Direct Memory Access
<b>RHEL</b>	Red Hat Enterprise Linux
<b>RoCE</b>	RDMA over Converged Ethernet
<b>SM</b>	Subnet Manager
<b>ToS</b>	Type of Service







REDP-5658-00

ISBN 0738460273

Printed in U.S.A.

Get connected

