TECHNICAL WHITE PAPER
September 2024

InfiniBand Configuration on VMware vSphere 8

vmware® by Broadcom

Contents

Introduction 3
Updated configuration workflow diagram 3
Steps to configure InfiniBand with vSphere 8.x 5
1 Make sure the native Mellanox driver exists in ESXi 5
2 Install MFT and NMST using vSphere Lifecycle Manager 5
2.1 Download the MFT and NMST packages 5
2.2 Import the MFT and NMST packages into Lifecycle Manager 5
2.3 Remediate hosts with Lifecycle Manager 6
2.4. Set the native Mellanox driver in recovery mode (optional) 7
3 Enable or disable InfiniBand SR-IOV 7
3.1 Enable SR-IOV 7
3.2 Disable SR-IOV if using passthrough 9
4 Configure InfiniBand switches 10
4.1 Update MLNX-OS on IB switches 10
4.2 Enable Open-SM virtualization support for SR-IOV 10
Known issues 11
Behavioral variations when passing through virtual and physical functions 11
Differences in Open-SM utilities 11
Differences in mst status 11
Troubleshooting for MLX cards not listed in the mst status command 12
Performance test 13
Unidirectional bandwidth 13
Bidirectional bandwidth 14
Conclusion 15
References 15
About the author 16
Acknowledgments 16

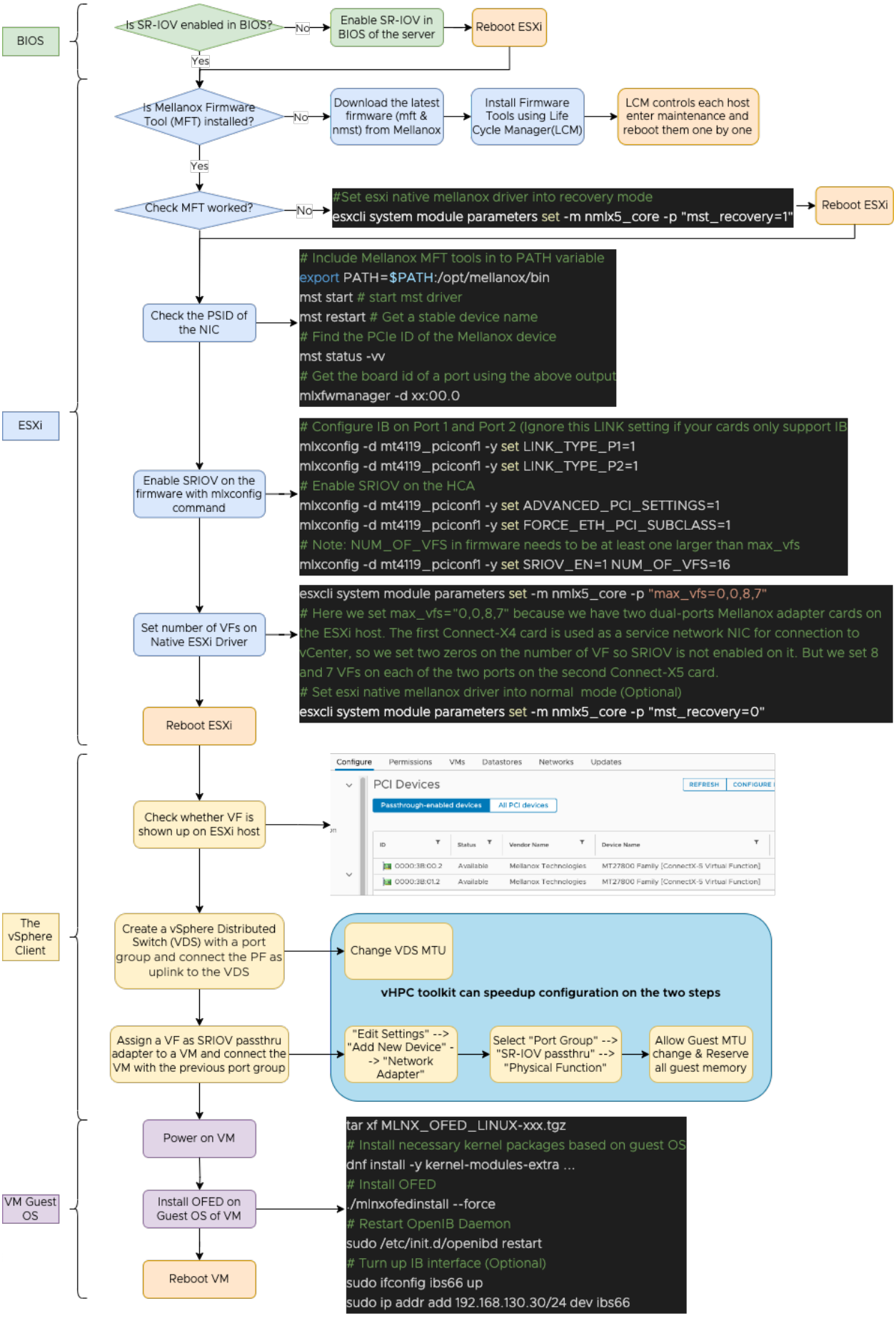
Introduction

Two years ago, we published technical white papers detailing the setup process for InfiniBand passthrough and InfiniBand SR-IOV on VMware vSphere® 7. In response to ongoing customer inquiries about implementing these features on vSphere 8 and VMware Cloud Foundation® (VCF), we are providing an update to our original paper.

Updated configuration workflow diagram

Figure 1 shows the updated workflow. The previous paper's workflow remains valid for the most part, but this update addresses a few key aspects that require special attention.

Figure 1. Flow chart to enable InfiniBand SR-IOV on NVIDIA/Mellanox ConnectX-5/6/7 in vSphere 8.x (next page)


BIOS
Is SR-IOV enabled in BIOS?
No
Enable SR-IOV in BIOS of the server
Reboot ESXi
Yes
Is Mellanox Firmware Tool (MFT) installed?
No
Download the latest firmware (mft & nmst) from Mellanox
Install Firmware Tools using Life Cycle Manager(LCM)
LCM controls each host enter maintenance and reboot them one by one
Yes
Check MFT worked?
No
#Set esxi native mellanox driver into recovery mode
esxcli system module parameters set -m nmlx5_core -p "mst_recovery=1"
Reboot ESXi
ESXi
Check the PSID of the NIC
Include Mellanox MFT tools in to PATH variable
export PATH=$PATH:/opt/mellanox/bin
mst start # start mst driver
mst restart # Get a stable device name
Find the PCIe ID of the Mellanox device
mst status -vv
Get the board id of a port using the above output
mlxfwmanager -d xx:00.0
Enable SRIOV on the firmware with mlxconfig command
Configure IB on Port 1 and Port 2 (Ignore this LINK setting if your cards only support IB
mlxconfig -d mt4119_pciconf1 -y set LINK_TYPE_P1=1
mlxconfig -d mt4119_pciconf1 -y set LINK_TYPE_P2=1
Enable SRIOV on the HCA
mlxconfig -d mt4119_pciconf1 -y set ADVANCED_PCI_SETTINGS=1
mlxconfig -d mt4119_pciconf1 -y set FORCE_ETH_PCI_SUBCLASS=1
Note: NUM_OF_VFS in firmware needs to be at least one larger than max_vfs
mlxconfig -d mt4119_pciconf1 -y set SRIOV_EN=1 NUM_OF_VFS=16
Set number of VFs on Native ESXi Driver
esxcli system module parameters set -m nmlx5_core -p "max_vfs=0,0,8,7"
Here we set max_vfs="0,0,8,7" because we have two dual-ports Mellanox adapter cards on the ESXi host. The first Connect-X4 card is used as a service network NIC for connection to vCenter, so we set two zeros on the number of VF so SRIOV is not enabled on it. But we set 8 and 7 VFs on each of the two ports on the second Connect-X5 card.
Set esxi native mellanox driver into normal mode (Optional)
esxcli system module parameters set -m nmlx5_core -p "mst_recovery=0"
Reboot ESXi
The vSphere Client
Check whether VF is shown up on ESXi host
Configure
Permissions
VMs
Datastores
Networks
Updates
PCI Devices
REFRESH
CONFIGURE
Passthrough-enabled devices
All PCI devices
ID
Status
Vendor Name
Device Name
0000:3B:00.2
Available
Mellanox Technologies
MT27800 Family [ConnectX-5 Virtual Function]
0000:3B:01.2
Available
Mellanox Technologies
MT27800 Family [ConnectX-5 Virtual Function]
Create a vSphere Distributed Switch (VDS) with a port group and connect the PF as uplink to the VDS
Change VDS MTU
vHPC toolkit can speedup configuration on the two steps
Assign a VF as SRIOV passthru adapter to a VM and connect the VM with the previous port group
"Edit Settings" --> "Add New Device" --> "Network Adapter"
Select "Port Group" --> "SR-IOV passthru" --> "Physical Function"
Allow Guest MTU change & Reserve all guest memory
VM Guest OS
Power on VM
Install OFED on Guest OS of VM
tar xf MLNX_OFED_LINUX-xxx.tgz
Install necessary kernel packages based on guest OS
dnf install -y kernel-modules-extra ...
Install OFED
./mlnxofedinstall --force
# Restart OpenIB Daemon
sudo /etc/init.d/openibd restart
# Turn up IB interface (Optional)
sudo ifconfig ibs66 up
sudo ip addr add 192.168.130.30/24 dev ibs66
Reboot VM

Steps to configure InfiniBand with vSphere 8.x

1 Make sure the native Mellanox driver exists in ESXi

vSphere 8.x includes a native Mellanox driver for VMware® ESXi™ hosts, eliminating the need for you to download it. To verify the presence of the driver, you can run the following command:

```
$ esxcli software vib list | grep -i nmlx5
nmlx5-cc            4.23.0.66-2vmw.802.0.0.22380479    VMW   VMwareCertified  2024-05-14  host
nmlx5-core          4.23.0.66-2vmw.802.0.0.22380479    VMW   VMwareCertified  2024-05-14  host
nmlx5-rdma          4.23.0.66-2vmw.802.0.0.22380479    VMW   VMwareCertified  2024-05-14  host
```

2 Install MFT and NMST using vSphere Lifecycle Manager

Installing Mellanox Firmware Tools (MFT and NMST) is now easier than ever with vSphere Lifecycle Manager. Gone are the days of using the esxcli command line on each host. Instead, virtual infrastructure () admins can leverage Lifecycle Manager to streamline the process.

2.1 Download the MFT and NMST packages

To get started, download the Mellanox Firmware Tools (MFT and NMST) packages from the following link: https://network.nvidia.com/products/adapter-software/firmware-tools/

Ensure you select the package compatible with vSphere 8.x. On the NVIDIA download page, this is **VMware ESX Server version 8.0 Native**.

2.2 Import the MFT and NMST packages into Lifecycle Manager

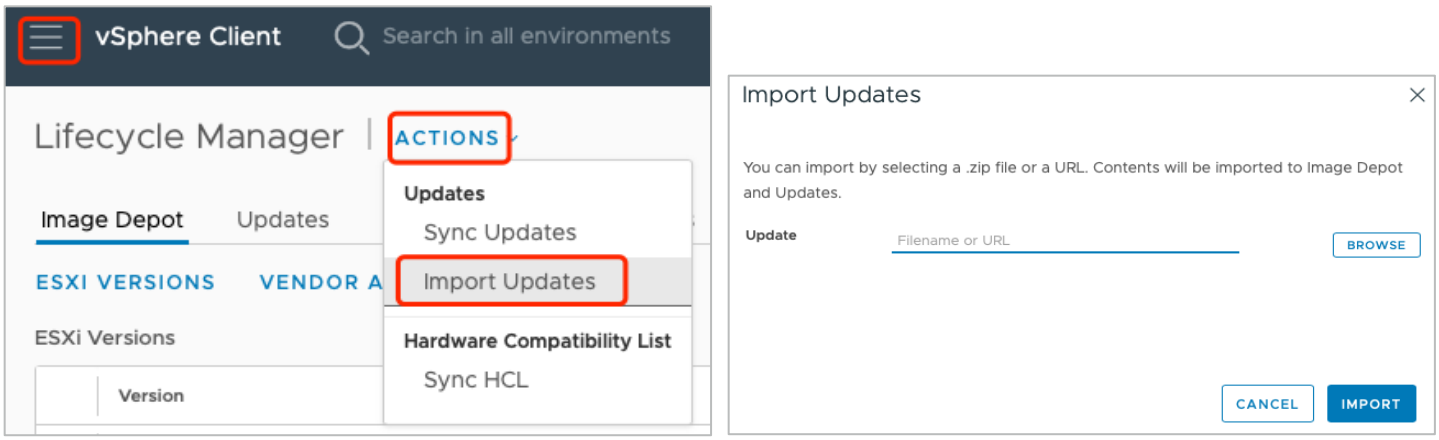
Once downloaded, **unpack** the packages and upload the following **zip** files to Lifecycle Manager:

- Mellanox-MFT-Tools_xxx.zip
- Mellanox-NATIVE-NMST_xxx.zip

To do this, follow these steps:

1. Open the vSphere Client menu by clicking on the three lines in the top left corner next to **vSphere Client**.
2. Select **Lifecycle Manager**.
3. Select **ACTIONS > Import Updates**.

Figure 2. Use Lifecycle Manager to import MFT and MST components

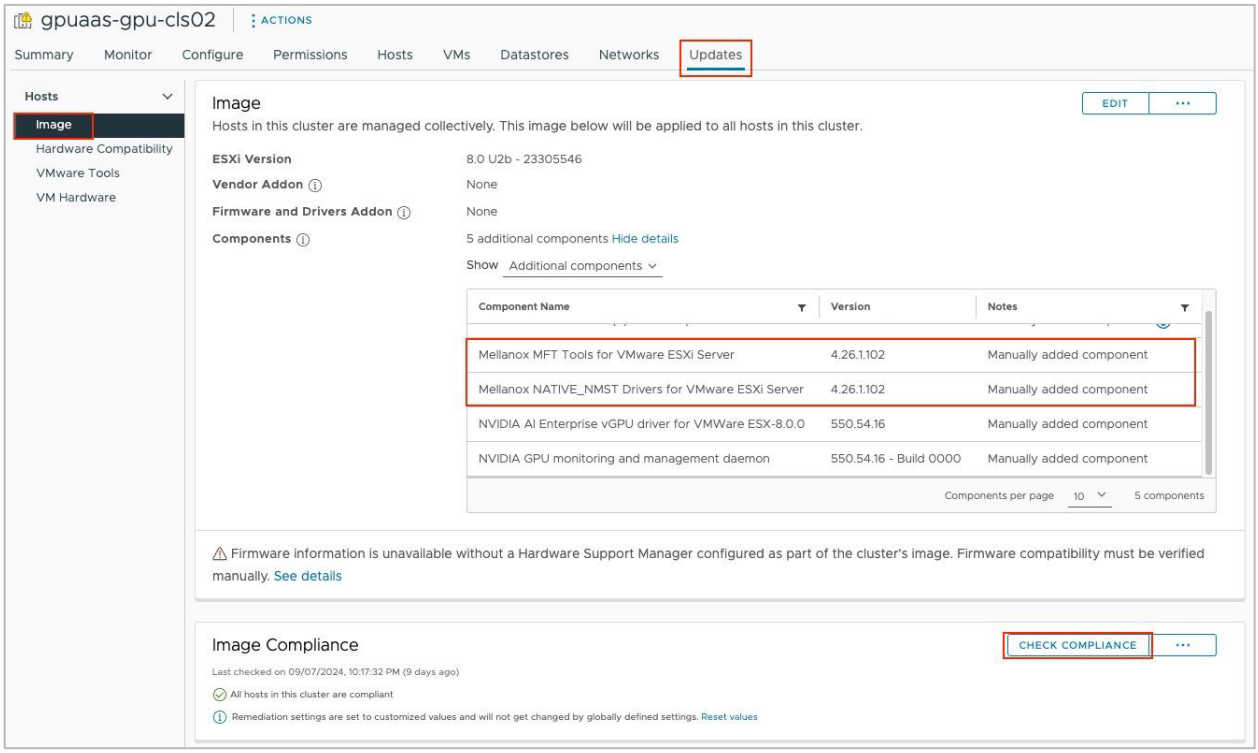


2.3 Remediate hosts with Lifecycle Manager

After importing the MFT packages, on the left panel, navigate to **Hosts > Updates** and click the **Images** tab in your targeted cluster. From there, you can:

- Check compliance to ensure all hosts are eligible for the update.
- Start the remediation process by following the guide: [Firmware Updates with vSphere Lifecycle Manager]

Figure 3. Use Lifecycle Manager to remediate the whole cluster



You can sit back and relax while Lifecycle Manager applies the update to each host in the cluster. Lifecycle Manager will automatically put each host into maintenance mode and install the components individually, ensuring a seamless and efficient update process.

2.4. Set the native Mellanox driver in recovery mode (optional)

In some cases, after installing the MFT and MST components, the IB cards may not be recognizable using the `mst status` command. This is usually due to the IB cards requiring a reset after the ESXi installation.

To resolve this issue:

Note: This process requires two reboots of the host: one to enable recovery mode and another to revert to normal mode. This operation typically needs a one-time effort.

1. Set the native Mellanox driver in recovery mode using the following command:

```
# Enabling Recovery Mode
esxcli system module parameters set -m nmlx5_core -p "mst_recovery=1"
```

2. After enabling recovery mode, reboot the ESXi host.
3. After the host is back online, you should see the IB devices listed in the `mst status` output.
4. After the IB devices are functioning correctly, you must set the IB native driver back to normal mode. To do this, run the following command:

```
# Reverting to Normal Mode
esxcli system module parameters set -m nmlx5_core -p "mst_recovery=0"
```

5. Reboot the ESXi host.

3 Enable or disable InfiniBand SR-IOV

If you have multiple IB cards installed on your ESXi host, especially for large language model (LLM) training, you'll need to configure SR-IOV.

Note: If you're using passthrough and plan to continue using it, you don't need to configure SR-IOV. If you have already enabled SR-IOV, you must disable it as described in step 3.2.

3.1 Enable SR-IOV

For hosts equipped with multiple IB cards, especially those used for large language model (LLM) training, it's essential to configure the IB cards to be recognized by the native mlx driver in ESXi. This involves setting the IB cards to be ETH controllers. To do this, run the following script on the ESXi host.

```
# Enable SRIOV
for i in 0 1 2 3 4 5 6 7
do
       device="mt4129_pciconf$i"
       echo "Configuring $device"

       mlxconfig -d $device -y set ADVANCED_PCI_SETTINGS=1
       mlxconfig -d $device -y set FORCE_ETH_PCI_SUBCLASS=1
       mlxconfig -d $device -y set SRIOV_EN=1 NUM_OF_VFS=16
done

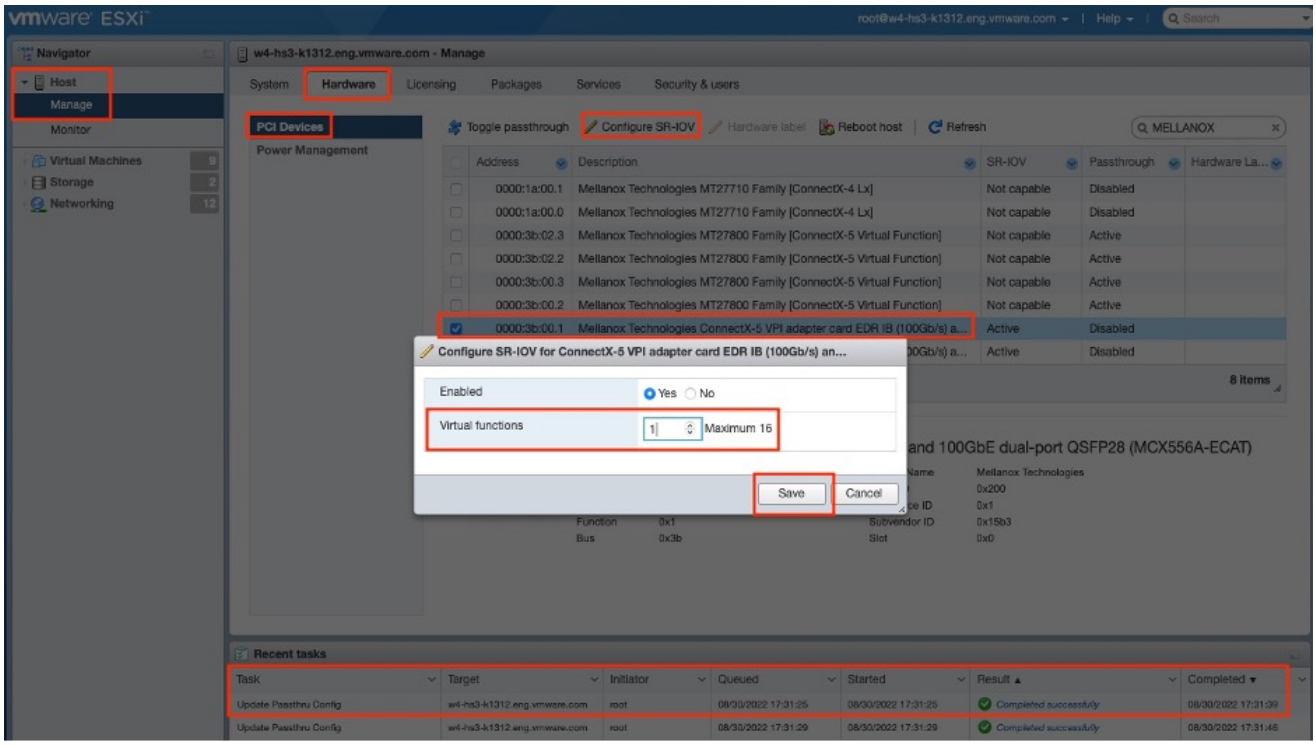
# Set ESXi module parameters
esxcli system module parameters set -m nmlx5_core -p "max_vfs=15,15,15,15,15,15,15,15"
```

The script uses three `mlxconfig` commands to configure Mellanox devices by modifying specific firmware settings.

- `mlxconfig -d $device -y set ADVANCED_PCI_SETTINGS=1`
  - Enables advanced PCIe configuration settings, allowing modification of low-level PCIe configurations.
- `mlxconfig -d $device -y set FORCE_ETH_PCI_SUBCLASS=1`
  - Forces the PCIe subclass to be recognized as an Ethernet controller. The underlying protocol is still the IB fabric.
- `mlxconfig -d $device -y set SRIOV_EN=1 NUM_OF_VFS=16`
  - Enables SR-IOV and creates 16 virtual functions (VFs) as an example. This number should be at least one larger than the sum of VFs created by max_vfs to reserve one VF for the physical function (PF).

Important notes:

- In vSphere 8.x, IB VFs are expected to display as **Down** in the UI, similar to vSphere 7. However, this does not indicate a problem. To verify the status of the IB VF, check the VM level for an active link state, which confirms that the VF is functioning correctly.
- The `esxcli` command is equivalent to configuring SR-IOV in the PCI Devices panel and setting the number of VFs in vCenter.
- **If you encounter issues, you can log into the ESXi host client to enable SR-IOV on the specific device directly.**



- If you encounter issues (e.g., changing the number of VFs is not responding in the vSphere Client), you can log into the ESXi host UI followed by **Host** → **Manage** → **Hardware** → **PCI Devices** → **Configure SR-IOV** → Select the physical adapters in the list → Change the **Number of virtual functions**.

3.2 Disable SR-IOV if using passthrough

If you need to disable SR-IOV to switch to passthrough mode, run the following script:

```
# Disable SR-IOV
for i in 0 1 2 3 4 5 6 7
do
    device="mt4129_pciconf$i"
    echo "Configuring $device"

    mlxconfig -d $device -y set ADVANCED_PCI_SETTINGS=0
    mlxconfig -d $device -y set SRIOV_EN=0
done

# Set ESXi module parameters
esxcli system module parameters set -m nmlx5_core -p ""
```

4 Configure InfiniBand switches

4.1 Update MLNX-OS on IB switches

If you're using an older MLNX-OS on your IB switches, contact your MLX support for assistance in updating to the latest version. Mismatched MLNX-OS versions in the switch and firmware versions in the MLX adapters can cause communication issues, including the IB interface showing as down in the VM. To verify your current MLNX-OS version, use the following command:

```
sc2-03-r04ibswa [standalone: master] (config) # show version
Product name:      MLNX-OS
Product release:   3.11.1004
Build ID:          #1-dev
Build date:        2023-08-09 11:05:30
Target arch:       x86_64
Target hw:         x86_64
Built by:          sw-r2d2-bot@3fe32015ceb9
Version summary:   X86_64 3.11.1004 2023-08-09 11:05:30 x86_64
```

4.2 Enable Open-SM virtualization support for SR-IOV

To ensure IB SR-IOV works correctly, verify that Open-SM virtualization support is enabled on the 9700 switch. Follow these steps:

```
sc2-03-r04ibswa [standalone: master] > en
sc2-03-r04ibswa [standalone: master] # config t
sc2-03-r04ibswa [standalone: master] (config) # show ib sm
enable
sc2-03-r04ibswa [standalone: master] (config) # show ib sm virt
Enabled
```

If Open-SM virtualization support is not enabled, run the following command to enable it:

```
ib sm virt enable
```

Known issues

Behavioral variations when passing through virtual and physical functions

Differences in Open-SM utilities

When passing a VF through to a VM, you might notice that the VFs cannot function in the output of the `sminfo` and `ibnodes` commands as follows. This behavior is expected because the output is normal when passing a VF through to a VM.

```
$ sudo sminfo
ibwarn: [263849] _do_madrpc: send failed; Operation not permitted
ibwarn: [263849] mad_rpc: _do_madrpc failed; dport (Lid 1)
sminfo: iberror: failed: quere

$ sudo ibhosts
ibwarn: [263855] _do_madrpc: send failed; Invalid argument
ibwarn: [263855] mad_rpc: _do_madrpc failed; dport (DR path slid 0; dlid 0; 0)
./libibnetdisc/ibnetdisc.c:807; Failed to resolve self
/usr/sbin/ibnetdiscover: iberror: failed: discover failed
```

In contrast, when passing a PF through to the VM, the commands should work as expected because Open-SM utilities function with PFs only.

```
$ sudo sminfo
sminfo: sm lid 1 sm guid 0xfc6a1c0300635dc0, activity count 1328 priority 0 state 3 SMINFO_MASTER

$ sudo ibnodes
Ca : 0xa088c2030006d8c0 ports 1 "MT4129 ConnectX7  Mellanox Technologies"
...
Switch : 0xfc6a1c0300635dc0 ports 65 "MF0;sc2-03-r04ibswb:MQM9700/U1" enhanced port 0 lid 1 lmc 0
```

Differences in mst status

When passing a PF through to the ESXi host, you can see the devices listed in the **Passthrough-enabled** tab in the **PCI Devices** section of the vSphere Client/vCenter. However, running the `mst status` command in the ESXi shell might not show these devices. This is expected in this use case because you would typically manage the devices' firmware at the VM level rather than the ESXi level.

Troubleshooting for MLX cards not listed in the mst status command

In some very rare cases, the nmlx5_core module may unexpectedly unload. To troubleshoot this issue, check whether the nmlx5_core module is loaded and enabled on your ESXi host by running the following command:

```
$ esxcli system module list | grep nmlx
nmlx5_core                             true        true
```

If the module is not loaded, you can reload it using the following steps:

1. Unload the nmlx5_rdma and nmlx5_core modules:

```
 > vmkload_mod -u nmlx5_rdma
 > vmkload_mod -u nmlx5_core
```

2. Load the nmlx5_core module with the mst_recovery=1 parameter:

```
 > vmkload_mod nmlx5_core mst_recovery=1
```

3. Check the status of the devmgr process and send a SIGHUP signal to the process.

```
 > ps -c | grep devmgr
 > kill -HUP <devmgr ps id>
```

4. Check if the card shows up when `mst status` is run.

After completing these steps, the nmlx5_core module will be in recovery mode.

5. Now, check if the card shows up when running the `mst status` command again.

Note: Since we don't use esxcli in the above commands, the recovery mode won't persist, thus the next reboot of the host will automatically switch the driver back to its normal mode.

Performance test

Unidirectional bandwidth

When using four queue pairs (qps), we reached a bandwidth of 396.5 gigabits per second (Gbps), nearly reaching the 400Gbps line rate of the IB cards.

```
ib_send_bw -a -q 4 --report_gbits 192.168.130.10
---------------------------------------------------------------------------------------
                    Send BW Test
 Dual-port       : OFF          Device         : mlx5_0
 Number of qps   : 4        Transport type : IB
 Connection type : RC         Using SRQ      : OFF
 PCIe relax order: ON
 ibv_wr* API     : ON
 TX depth        : 128
 CQ Moderation   : 100
 Mtu             : 4096[B]
 Link type       : IB
 Max inline data : 0[B]
 rdma_cm QPs     : OFF
 Data ex. method : Ethernet
---------------------------------------------------------------------------------------
 local address: LID 0x23 QPN 0x0129 PSN 0xc88dff
 local address: LID 0x23 QPN 0x012b PSN 0xd2f97d
 local address: LID 0x23 QPN 0x012c PSN 0xa80983
 local address: LID 0x23 QPN 0x012d PSN 0x25e16
 remote address: LID 0x22 QPN 0x0129 PSN 0x2b2a73
 remote address: LID 0x22 QPN 0x012b PSN 0x4fcee1
 remote address: LID 0x22 QPN 0x012c PSN 0xb6bd17
 remote address: LID 0x22 QPN 0x012d PSN 0xbcc1a
---------------------------------------------------------------------------------------
 #bytes     #iterations    BW peak[Gb/sec]    BW average[Gb/sec]   MsgRate[Mpps]
 2          4000             0.067797           0.066441            4.152563
 4          4000             0.14               0.14                4.253774
 8          4000             0.28               0.27                4.285626
 16         4000             0.55               0.55                4.283263
 32         4000             1.09               1.08                4.214406
 64         4000             2.20               2.19                4.283350
 128        4000             4.40               4.39                4.283552
 256        4000             8.71               8.70                4.247094
 512        4000             17.58              17.54               4.281443
 1024       4000             35.08              34.16               4.170244
 2048       4000             70.32              70.03               4.274256
 4096       4000             139.44             139.02              4.242477
 8192       4000             277.11             276.44              4.218121
 16384      4000             391.84             391.53              2.987155
 32768      4000             394.20             394.15              1.503556
 65536      4000             395.54             395.40              0.754171
 131072     4000             395.99             395.98              0.377632
 262144     4000             396.25             396.24              0.188944
 524288     4000             396.40             396.38              0.094506
 1048576    4000             396.46             396.45              0.047260
 2097152    4000             396.48             396.48              0.023632
 4194304    4000             396.49             396.49              0.011816
 8388608    4000             396.50             396.50              0.005908
---------------------------------------------------------------------------------------
```

Note: You may notice that the **Link type** is listed as **IB**, but the **Data ex. method** is listed as **Ethernet**. This is expected behavior because the native mlx ESXi driver recognizes the IB card as an Ethernet adapter.

```
$ ibstat
CA 'mlx5_0'
    CA type: MT4126
    Number of ports: 1
    Firmware version: 28.39.1002
    Hardware version: 0
    Node GUID: 0x005056fffeacc71f
    System image GUID: 0xa088c203001577f0
    Port 1:
        State: Active
        Physical state: LinkUp
        Rate: 400
        Base lid: 35
        LMC: 0
        SM lid: 1
        Capability mask: 0xa751ec48
        Port GUID: 0x005056fffeacc71f
        Link layer: InfiniBand
```

Bidirectional bandwidth

Similarly, we can launch the bidirectional IB perftest and observe that we nearly reach 790Gbps, nearly reaching the two IB cards' 800Gbps line rate (400*2=800).

```
$ ib_read_bw  -d mlx5_4 --report_gbit -a -b -F 192.168.130.10 --limit_bw=780 -q 4
---------------------------------------------------------------------------------------
                    RDMA_Read Bidirectional BW Test
 Dual-port       : OFF      Device         : mlx5_4
 Number of qps   : 4        Transport type : IB
 Connection type : RC       Using SRQ      : OFF
 PCIe relax order: ON
 ibv_wr* API     : ON
 TX depth        : 128
 CQ Moderation   : 100
 Mtu             : 4096[B]
 Link type       : IB
 Outstand reads  : 16
 rdma_cm QPs     : OFF
 Data ex. method : Ethernet
---------------------------------------------------------------------------------------
 local address: LID 0x03 QPN 0x0067 PSN 0xb35d1b OUT 0x10 RKey 0x1fff00 VAddr 0x007fc830073000
 local address: LID 0x03 QPN 0x0068 PSN 0xb05069 OUT 0x10 RKey 0x1fff00 VAddr 0x007fc830873000
 local address: LID 0x03 QPN 0x0069 PSN 0x8e13ff OUT 0x10 RKey 0x1fff00 VAddr 0x007fc831073000
 local address: LID 0x03 QPN 0x006a PSN 0xb03ce2 OUT 0x10 RKey 0x1fff00 VAddr 0x007fc831873000
 remote address: LID 0x15 QPN 0x0063 PSN 0x5a2ec3 OUT 0x10 RKey 0x1fff00 VAddr 0x007f2eec0c4000
 remote address: LID 0x15 QPN 0x0064 PSN 0xf484f1 OUT 0x10 RKey 0x1fff00 VAddr 0x007f2eec8c4000
 remote address: LID 0x15 QPN 0x0065 PSN 0x2aa1e7 OUT 0x10 RKey 0x1fff00 VAddr 0x007f2eed0c4000
 remote address: LID 0x15 QPN 0x0066 PSN 0xb718aa OUT 0x10 RKey 0x1fff00 VAddr 0x007f2eed8c4000
---------------------------------------------------------------------------------------
```

```
#bytes     #iterations    BW peak[Gb/sec]    BW average[Gb/sec]   MsgRate[Mpps]
 2           4000             0.132252           0.130733           8.170815
 4           4000             0.26               0.26               8.217808
 8           4000             0.53               0.53               8.268855
 16          4000             1.05               1.04               8.157475
 32          4000             2.12               2.09               8.153349
 64          4000             4.23               4.22               8.232741
 128         4000             8.48               8.45               8.255226
 256         4000             16.93              16.80              8.204786
 512         4000             33.75              33.62              8.208010
 1024        4000             67.64              67.40              8.226943
 2048        4000             133.95             133.05             8.120904
 4096        4000             267.28             265.24             8.094578
 8192        4000             528.59             526.79             8.038249
 16384       4000             765.39             765.01             5.836597
 32768       4000             780.19             779.65             2.974111
 65536       4000             786.19             786.03             1.499235
 131072      4000             787.66             787.59             0.751109
 262144      4000             789.66             789.59             0.376507
 524288      4000             789.65             789.63             0.188263
 1048576     4000             789.83             789.82             0.094154
 2097152     4000             790.05             790.04             0.047090
 4194304     4000             790.21             790.20             0.023550
 8388608     4000             790.20             790.20             0.011775
---------------------------------------------------------------------------------------
```

Conclusion

We updated our original technical paper for InfiniBand configuration and performance on vSphere 8.x. We covered the updated configuration workflow, simplified installation of MFT and NMST using vSphere Lifecycle Manager, and IB SR-IOV configuration. We also discussed known issues and provided performance results for unidirectional and bidirectional IB perftest. With the updated configuration and performance results, you can now take full advantage of InfiniBand on vSphere 8.x and VMware Cloud Foundation (VCF).

References

[1] InfiniBand/RoCE Setup and Performance on vSphere 7.x

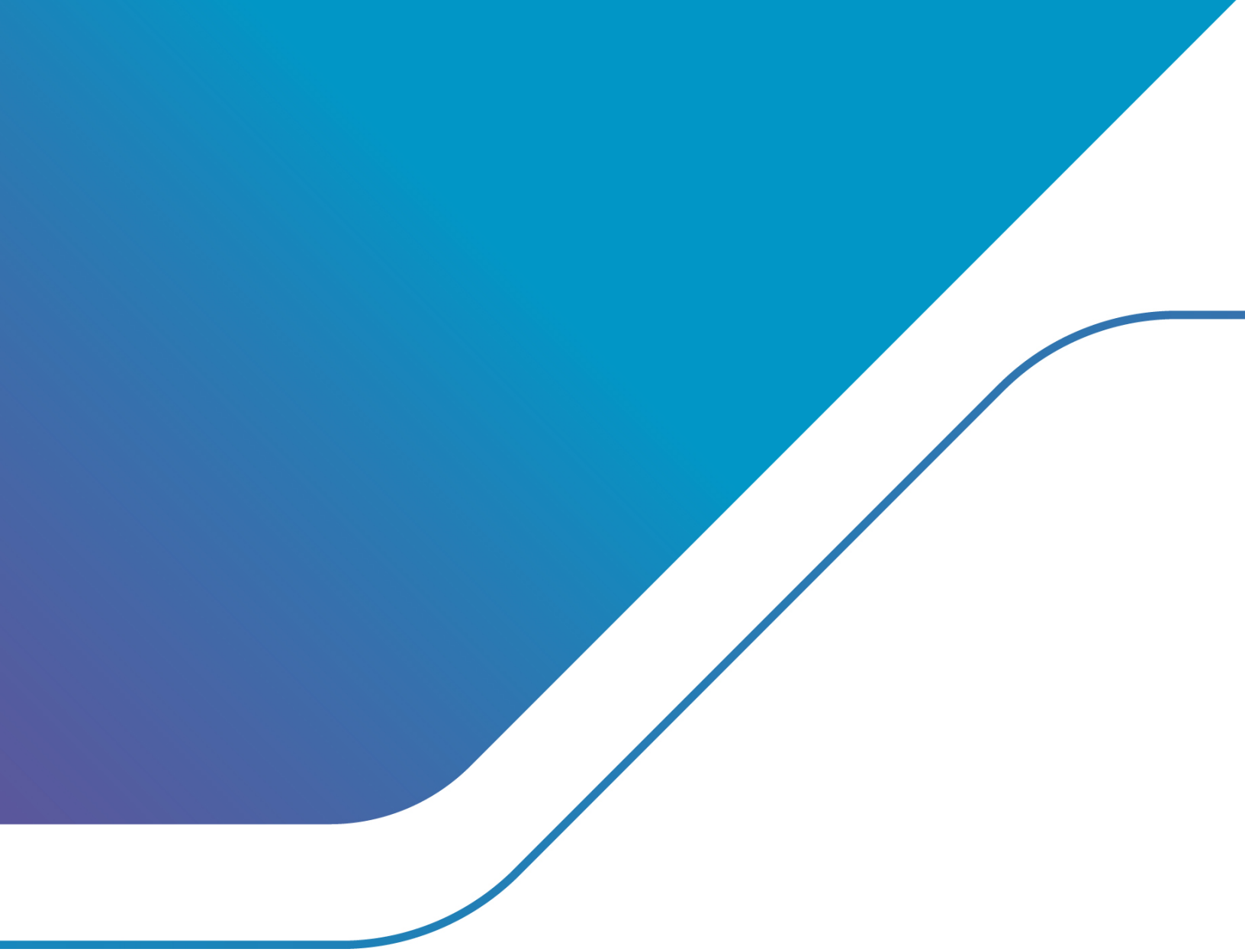
[2] InfiniBand SR-IOV Setup and Performance Study on vSphere 7.x

About the author

Yuankun Fu has been a seasoned software engineer in the AI Platform and Solution team within the VCF division at VMware and Broadcom since July 2021. With over 13 years of experience in HPC, Yuankun focuses on optimizing HPC and AI application performance on the VMware platform. His work encompasses a broad range of projects, including technical guides, performance best practices, and troubleshooting complex performance issues that arise when running demanding workloads on customer platforms. Prior to joining VMware, Yuankun earned his PhD in Computer Science from Purdue University and interned at the Los Alamos National Lab.

Acknowledgments

The author thanks Ramesh Radhakrishnan and Julie Brodeur from VMware for their support in editing and improving the paper.



vmware® by Broadcom

Copyright © 2024 Broadcom. All rights reserved.
The term "Broadcom" refers to Broadcom Inc. and/or its subsidiaries. For more information, go to www.broadcom.com. All trademarks, trade names, service marks, and logos referenced herein belong to their respective companies. Broadcom reserves the right to make changes without further notice to any products or data herein to improve reliability, function, or design. Broadcom believes the Information herein is accurate and reliable. However, Broadcom does not assume any liability arising out of the application or use of this information, or the application or use of any product described herein. It does not convey any license under its patent rights or the rights of others.