DOCA Infrastructure

Table of contents

Preface

TBD

Command Cheat Sheet

TBD

Logging and Counters

TBD

Debug Info Packages

TBD

Scenarios

RShim Troubleshooting and How-Tos

Another backend already attached

Several generations of BlueField DPUs are equipped with a USB interface in which RShim can be routed, via USB cable, to an external host running Linux and the RShim driver.

In this case, typically following a system reboot, the RShim over USB prevails and the DPU host reports RShim status as "`another backend already attached`". This is correct behavior, since there can only be one RShim backend active at any given time. However, this means that the DPU host does not own RShim access.

To reclaim RShim ownership safely:

1. Stop the RShim driver on the remote Linux. Run:

```
systemctl stop rshim
systemctl disable rshim
```

2. Restart RShim on the DPU host. Run:

```
systemctl enable rshim
systemctl start rshim
```

The "`another backend already attached`" scenario can also be attributed to the RShim backend being owned by the BMC in DPUs with integrated BMC. This is elaborated on further down on this page.

RShim driver not loading

Verify whether your DPU features an integrated BMC or not. Run:

```
# sudo sudo lspci -s $(sudo lspci -d 15b3: | head -1 | awk
'{print $1}') -vvv | grep "Product Name"
```

Example output for DPU **with integrated BMC**:

```
Product Name: BlueField-2 DPU 25GbE Dual-Port SFP56, integrated
BMC, Crypto and Secure Boot Enabled, 16GB on-board DDR, 1GbE OOB
management, Tall Bracket, FHHL
```

If your DPU has an integrated BMC, refer to RShim driver not loading on host with integrated BMC.

If your DPU does not have an integrated BMC, refer to RShim driver not loading on host on DPU without integrated BMC.

RShim driver not loading on DPU with integrated BMC

RShim driver not loading on host

1. Access the BMC via the RJ45 management port of the DPU.
2. Delete RShim on the BMC:

```
systemctl stop rshim
systemctl disable rshim
```

3. Enable RShim on the host:

```
systemctl enable rshim
systemctl start rshim
```

4. Restart RShim service. Run:

```
sudo systemctl restart rshim
```

   If RShim service does not launch automatically, run:

```
sudo systemctl status rshim
```

   This command is expected to display "`active (running)`".

5. Display the current setting. Run:

```
# cat /dev/rshim<N>/misc | grep DEV_NAME
DEV_NAME        pcie-0000:04:00.2
```

   This output indicates that the RShim service is ready to use.

RShim driver not loading on BMC

1. Verify that the RShim service is not running on host. Run:

   ```
   systemctl status rshim
   ```

   If the output is `active`, then it may be presumed that the host has ownership of the RShim.

2. Delete RShim on the host. Run:

   ```
   systemctl stop rshim
   systemctl disable rshim
   ```

3. Enable RShim on the BMC. Run:

   ```
   systemctl enable rshim
   systemctl start rshim
   ```

4. Display the current setting. Run:

   ```
   # cat /dev/rshim<N>/misc | grep DEV_NAME
   DEV_NAME        usb-1.0
   ```

   This output indicates that the RShim service is ready to use.

RShim driver not loading on host on DPU without integrated BMC

1. Download the suitable DEB/RPM for RShim (management interface for DPU from the host) driver.
2. Reinstall RShim package on the host.
    - For Ubuntu/Debian, run:

      ```
      sudo dpkg --force-all -i rshim-<version>.deb
      ```

    - For RHEL/CentOS, run:

      ```
      sudo rpm -Uhv rshim-<version>.rpm
      ```

3. Restart RShim service. Run:

   ```
   sudo systemctl restart rshim
   ```

   If RShim service does not launch automatically, run:

   ```
   sudo systemctl status rshim
   ```

   This command is expected to display "`active (running)`".

4. Display the current setting. Run:

   ```
   # cat /dev/rshim<N>/misc | grep DEV_NAME
   DEV_NAME        pcie-0000:04:00.2
   ```

   This output indicates that the RShim service is ready to use.

Change ownership of RShim from NIC BMC to host

1. Verify that your card has BMC. Run the following on the host:

   ```
   # sudo sudo lspci -s $(sudo lspci -d 15b3: | head -1 | awk '{print $1}') -vvv |grep "Product Name"
   Product Name: BlueField-2 DPU 25GbE Dual-Port SFP56, integrated BMC, Crypto and Secure Boot Enabled, 16GB on-board DDR, 1GbE OOB management, Tall Bracket, FHHL
   ```

   The product name is supposed to show "integrated BMC".

2. Access the BMC via the RJ45 management port of the DPU.

3. Delete RShim on the BMC:

   ```
   systemctl stop rshim
   systemctl disable rshim
   ```

4. Enable RShim on the host:

   ```
   systemctl enable rshim
   systemctl start rshim
   ```

5. Restart RShim service. Run:

   ```
   sudo systemctl restart rshim
   ```

If RShim service does not launch automatically, run:

```
sudo systemctl status rshim
```

This command is expected to display "`active (running)`".

6. Display the current setting. Run:

```
# cat /dev/rshim<N>/misc | grep DEV_NAME
DEV_NAME        pcie-0000:04:00.2
```

This output indicates that the RShim service is ready to use.

Connectivity Troubleshooting

Connection (ssh, screen console) to the DPU is lost

The UART cable in the Accessories Kit (OPN: MBF20-DKIT) can be used to connect to the DPU console and identify the stage at which BlueField is hanging.

Follow this procedure:

1. Connect the UART cable to a USB socket, and find it in your USB devices.

```
sudo lsusb
Bus 002 Device 003: ID 0403:6001 Future Technology Devices
International, Ltd FT232 Serial (UART) IC
```

Note

For more information on the UART connectivity, please refer to the DPU's hardware user guide under Supported Interfaces > Interfaces Detailed Description > NC-SI Management Interface.

Info

It is good practice to connect the other end of the NC-SI cable to a different host than the one on which the BlueField DPU is installed.

2. Install the minicom application.

OS	Command
CentOS/RHEL	`sudo yum install minicom -y`
Ubuntu/Debian	`sudo apt-get install minicom`

3. Open the minicom application.

```
sudo minicom -s -c on
```

4. Go to "Serial port setup".

5. Enter "F" to change "Hardware Flow control" to NO.

6. Enter "A" and change to `/dev/ttyUSB0` and press Enter.

7. Press ESC.

8. Type on "Save setup as dfl".

9. Exit minicom by pressing Ctrl + a + z.

Driver not loading in host server

What this looks like in dmsg:

```
[275604.216789] mlx5_core 0000:af:00.1: 63.008 Gb/s available
PCIe bandwidth, limited by 8 GT/s x8 link at 0000:ae:00.0
(capable of 126.024 Gb/s with 16 GT/s x8 link)
[275624.187596] mlx5_core 0000:af:00.1: wait_fw_init:316:(pid
943): Waiting for FW initialization, timeout abort in 100s
[275644.152994] mlx5_core 0000:af:00.1: wait_fw_init:316:(pid
943): Waiting for FW initialization, timeout abort in 79s
[275664.118404] mlx5_core 0000:af:00.1: wait_fw_init:316:(pid
943): Waiting for FW initialization, timeout abort in 59s
[275684.083806] mlx5_core 0000:af:00.1: wait_fw_init:316:(pid
943): Waiting for FW initialization, timeout abort in 39s
[275704.049211] mlx5_core 0000:af:00.1: wait_fw_init:316:(pid
943): Waiting for FW initialization, timeout abort in 19s
[275723.954752] mlx5_core 0000:af:00.1: mlx5_function_setup:1237:
(pid 943): Firmware over 120000 MS in pre-initializing state,
aborting
[275723.968261] mlx5_core 0000:af:00.1: init_one:1813:(pid 943):
mlx5_load_one failed with error code -16
[275723.978578] mlx5_core: probe of 0000:af:00.1 failed with
error -16
```

The driver on the host server is dependent on the Arm side. If the driver on Arm is up, then the driver on the host server will also be up.

Please verify that:

- The driver is loaded in the BlueField DPU

- The Arm is booted into OS
- The Arm is not in UEFI Boot Menu
- The Arm is not hanged

Then:

1. Perform graceful shutdown.
2. Power cycle on the host server.
3. If the problem persists, reset nvconfig (`sudo mlxconfig -d /dev/mst/<device> -y reset`) and power cycle the host.

   > **Note**
   >
   > If your DPU is VPI capable, please be aware that this configuration will reset the link type on the network ports to IB. To change the network port's link type to Ethernet, run:
   >
   > ```
   > sudo mlxconfig -d <device> s LINK_TYPE_P1=2 LINK_TYPE_P2=2
   > ```

4. If this problem persists, please make sure to install the latest bfb image and then restart the driver in host server. Please refer to this page for more information.

No connectivity between network interfaces of source host to destination device

Verify that the bridge is configured properly on the Arm side.

The following is an example for default configuration:

```
$ sudo ovs-vsctl show
f6740bfb-0312-4cd8-88c0-a9680430924f
    Bridge ovsbr1
        Port pf0sf0
            Interface pf0sf0
        Port p0
            Interface p0
        Port pf0hpf
            Interface pf0hpf
        Port ovsbr1
            Interface ovsbr1
                type: internal
    Bridge ovsbr2
        Port p1
            Interface p1
        Port pf1sf0
            Interface pf1sf0
        Port pf1hpf
            Interface pf1hpf
        Port ovsbr2
            Interface ovsbr2
                type: internal
    ovs_version: "2.14.1"
```

If no bridge configuration exists, refer to "Virtual Switch on DPU".

Uplink in Arm down while uplink in host server up

Please check that the cables are connected properly into the network ports of the DPU and the peer device.

Performance Degradation

Degradation in performance indicates that openvswitch may not be offloaded.

Verify offload state. Run:

```
# ovs-vsctl get Open_vSwitch . other_config:hw-offload
```

- If `hw-offload = true` – Fast Pass is configured (desired result)
- If `hw-offload = false` – Slow Pass is configured

If `hw-offload = false`:

- For RHEL/CentOS, run:

  ```
  # ovs-vsctl set Open_vSwitch . other_config:hw-offload=true;
  # systemctl restart openvswitch;
  # systemctl enable openvswitch;
  ```

- For Ubuntu/Debian, run:

  ```
  # ovs-vsctl set Open_vSwitch . other_config:hw-offload=true;
  # /etc/init.d/openvswitch-switch restart
  ```

SR-IOV Troubleshooting

Unable to create VFs

1. Please make sure that SR-IOV is enabled in BIOS.

2. Verify `SRIOV_EN` is true and `NUM_OF_VFS` bigger than 1. Run:

```
# mlxconfig -d /dev/mst/mt41686_pciconf0 -e q |grep -i
"SRIOV_EN\|num_of_vf"
Configurations:          Default         Current
Next Boot
*        NUM_OF_VFS      16              16              16
*        SRIOV_EN        True(1)         True(1)
True(1)
```

3. Verify that `GRUB_CMDLINE_LINUX="iommu=pt intel_iommu=on pci=assign-busses"`.

No traffic between VF to external host

1. Please verify creation of representors for VFs inside the Bluefield DPU. Run:

```
# /opt/mellanox/iproute2/sbin/rdma link |grep -i up
...
link mlx5_0/2 state ACTIVE physical_state LINK_UP netdev
pf0vf0
...
```

2. Make sure the representors of the VFs are added to the bridge. Run:

```
# ovs-vsctl add-port <bridage_name> pf0vf0
```

3. Verify VF configuration. Run:

```
$ ovs-vsctl show
bb993992-7930-4dd2-bc14-73514854b024
    Bridge ovsbr1
        Port pf0vf0
            Interface pf0vf0
                type: internal
        Port pf0hpf
            Interface pf0hpf
        Port pf0sf0
            Interface pf0sf0
        Port p0
            Interface p0
    Bridge ovsbr2
        Port ovsbr2
            Interface ovsbr2
                type: internal
        Port pf1sf0
            Interface pf1sf0
        Port p1
            Interface p1
        Port pf1hpf
            Interface pf1hpf
    ovs_version: "2.14.1"
```

eSwitch Troubleshooting

Unable to configure legacy mode

To set devlink to "Legacy" mode in BlueField, run:

```
# devlink dev eswitch set pci/0000:03:00.0 mode legacy
```

```
# devlink dev eswitch set pci/0000:03:00.1 mode legacy
```

Please verify that:

- No virtual functions are open. To verify if VFs are configured, run:

  ```
  # /opt/mellanox/iproute2/sbin/rdma link | grep -i up
  link mlx5_0/2 state ACTIVE physical_state LINK_UP netdev
  pf0vf0
  link mlx5_1/2 state ACTIVE physical_state LINK_UP netdev
  pf1vf0
  ```

  If any VFs are configured, destroy them by running:

  ```
  # echo 0 > /sys/class/infiniband/mlx5_0/device/mlx5_num_vfs
  # echo 0 > /sys/class/infiniband/mlx5_1/device/mlx5_num_vfs
  ```

- If any SFs are configured, delete them by running:

  ```
  /sbin/mlnx-sf -a delete --sfindex <SF-Index>
  ```

  Note

  You may retrieve the `<SF-Index>` of the currently installed SFs by running:

  ```
  # mlnx-sf -a show

  SF Index: pci/0000:03:00.0/229408
  ```

```
  Parent PCI dev: 0000:03:00.0
  Representor netdev: en3f0pf0sf0
  Function HWADDR: 02:61:f6:21:32:8c
  Auxiliary device: mlx5_core.sf.2
    netdev: enp3s0f0s0
    RDMA dev: mlx5_2

SF Index: pci/0000:03:00.1/294944
  Parent PCI dev: 0000:03:00.1
  Representor netdev: en3f1pf1sf0
  Function HWADDR: 02:30:13:6a:2d:2c
  Auxiliary device: mlx5_core.sf.3
    netdev: enp3s0f1s0
    RDMA dev: mlx5_3
```

Pay attention to the SF Index values. For example:

```
/sbin/mlnx-sf -a delete --sfindex
pci/0000:03:00.0/229408
/sbin/mlnx-sf -a delete --sfindex
pci/0000:03:00.1/294944
```

If the error "`Error: mlx5_core: Can't change mode when flows are configured`" is encountered while trying to configure legacy mode, please make sure that

1. Any configured SFs are deleted (see above for commands).
2. Shut down the links of all interfaces, delete any `ip xfrm` rules, delete any configured OVS flows, and stop openvswitch service. Run:

```
ip link set dev p0 down
ip link set dev p1 down
```

```
ip link set dev pf0hpf down
ip link set dev pf1hpf down
ip link set dev vxlan_sys_4789 down

ip x s f ;
ip x p f ;

tc filter del dev p0 ingress
tc filter del dev p1 ingress
tc qdisc show dev p0
tc qdisc show dev p1
tc qdisc del dev p0 ingress
tc qdisc del dev p1 ingress
tc qdisc show dev p0
tc qdisc show dev p1

systemctl stop openvswitch-switch
```

DPU appears as two interfaces

What this looks like:

```
# sudo /opt/mellanox/iproute2/sbin/rdma link
link mlx5_0/1 state ACTIVE physical_state LINK_UP netdev p0
link mlx5_1/1 state ACTIVE physical_state LINK_UP netdev p1
```

- Check if you are working in legacy mode.

  ```
  # devlink dev eswitch show pci/0000:03:00.<0|1>
  ```

  If the following line is printed, this means that you are working in legacy mode:

```
pci/0000:03:00.<0|1>: mode legacy inline-mode none encap
enable
```

Please configure the DPU to work in switchdev mode. Run:

```
devlink dev eswitch set pci/0000:03:00.<0|1> mode switchdev
```

- Check if you are working in separated mode:

```
# mlxconfig -d /dev/mst/mt41686_pciconf0 q | grep -i cpu
* INTERNAL_CPU_MODEL SEPERATED_HOST(0)
```

Please configure the DPU to work in embedded mode. Run:

```
# mlxconfig -d /dev/mst/mt41686_pciconf0 s
INTERNAL_CPU_MODEL=1
```

Notice

This document is provided for information purposes only and shall not be regarded as a warranty of a certain functionality, condition, or quality of a product. NVIDIA Corporation (“NVIDIA”) makes no representations or warranties, expressed or implied, as to the accuracy or completeness of the information contained in this document and assumes no responsibility for any errors contained herein. NVIDIA shall have no liability for the consequences or use of such information or for any infringement of patents or other rights of third parties that may result from its use. This document is not a commitment to develop, release, or deliver any Material (defined below), code, or functionality.

NVIDIA reserves the right to make corrections, modifications, enhancements, improvements, and any other changes to this document, at any time without notice.

Customer should obtain the latest relevant information before placing orders and should verify that such information is current and complete.

NVIDIA products are sold subject to the NVIDIA standard terms and conditions of sale supplied at the time of order acknowledgement, unless otherwise agreed in an individual sales agreement signed by authorized representatives of NVIDIA and customer (“Terms of Sale”). NVIDIA hereby expressly objects to applying any customer general terms and conditions with regards to the purchase of the NVIDIA product referenced in this document. No contractual obligations are formed either directly or indirectly by this document.

NVIDIA products are not designed, authorized, or warranted to be suitable for use in medical, military, aircraft, space, or life support equipment, nor in applications where failure or malfunction of the NVIDIA product can reasonably be expected to result in personal injury, death, or property or environmental damage. NVIDIA accepts no liability for inclusion and/or use of NVIDIA products in such equipment or applications and therefore such inclusion and/or use is at customer’s own risk.

NVIDIA makes no representation or warranty that products based on this document will be suitable for any specified use. Testing of all parameters of each product is not necessarily performed by NVIDIA. It is customer’s sole responsibility to evaluate and determine the applicability of any information contained in this document, ensure the product is suitable and fit for the application planned by customer, and perform the necessary testing for the application in order to avoid a default of the application or the product. Weaknesses in customer’s product designs may affect the quality and reliability of the NVIDIA product and may result in additional or different conditions and/or requirements beyond those contained in this document. NVIDIA accepts no liability related to any default, damage, costs, or problem which may be based on or attributable to: (i) the use of the NVIDIA product in any manner that is contrary to this document or (ii) customer product designs.

No license, either expressed or implied, is granted under any NVIDIA patent right, copyright, or other NVIDIA intellectual property right under this document. Information published by NVIDIA regarding third-party products or services does not constitute a license from NVIDIA to use such products or services or a warranty or endorsement thereof. Use of such information may require a license from a third party under the patents or other intellectual property rights of the third party, or a license from NVIDIA under the patents or other intellectual property rights of NVIDIA.

Reproduction of information in this document is permissible only if approved in advance by NVIDIA in writing, reproduced without alteration and in full compliance with all applicable export laws and regulations, and accompanied by all associated conditions, limitations, and notices.

THIS DOCUMENT AND ALL NVIDIA DESIGN SPECIFICATIONS, REFERENCE BOARDS, FILES, DRAWINGS, DIAGNOSTICS, LISTS, AND OTHER DOCUMENTS (TOGETHER AND SEPARATELY, “MATERIALS”) ARE BEING PROVIDED “AS IS.” NVIDIA MAKES NO WARRANTIES, EXPRESSED, IMPLIED, STATUTORY, OR OTHERWISE WITH RESPECT TO THE MATERIALS, AND EXPRESSLY DISCLAIMS ALL IMPLIED WARRANTIES OF NONINFRINGEMENT, MERCHANTABILITY, AND FITNESS FOR A PARTICULAR PURPOSE. TO THE EXTENT NOT PROHIBITED BY LAW, IN NO EVENT WILL NVIDIA BE LIABLE FOR ANY DAMAGES, INCLUDING WITHOUT LIMITATION ANY DIRECT, INDIRECT, SPECIAL, INCIDENTAL, PUNITIVE, OR CONSEQUENTIAL DAMAGES, HOWEVER CAUSED AND REGARDLESS OF THE THEORY OF LIABILITY, ARISING OUT OF

ANY USE OF THIS DOCUMENT, EVEN IF NVIDIA HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES. Notwithstanding any damages that customer might incur for any reason whatsoever, NVIDIA's aggregate and cumulative liability towards customer for the products described herein shall be limited in accordance with the Terms of Sale for the product.

Trademarks

NVIDIA and the NVIDIA logo are trademarks and/or registered trademarks of NVIDIA Corporation in the U.S. and other countries. Other company and product names may be trademarks of the respective companies with which they are associated.