



VXLAN Hardware Stateless Offloads

Table of contents

Enabling VXLAN Hardware Stateless Offloads

Important Note

List of Figures

Figure 0. Procedure Heading Icon Version 1 Modificationdate
1717697200854 Api V2

VXLAN technology provides scalability and security challenges solutions. It requires extension of the traditional stateless offloads to avoid performance drop. ConnectX family cards offer the following stateless offloads for a VXLAN packet, similar to the ones offered to non-encapsulated packets. VXLAN protocol encapsulates its packets using outer UDP header.

Available hardware stateless offloads:

- Checksum generation (Inner IP and Inner TCP/UDP)
- Checksum validation (Inner IP and Inner TCP/UDP)
- TSO support for inner TCP packets
- RSS distribution according to inner packets attributes
- Receive queue selection - inner frames may be steered to specific QPs

Enabling VXLAN Hardware Stateless Offloads

VXLAN offload is enabled by default for ConnectX-4 family devices running the minimum required firmware version and a kernel version that includes VXLAN support.

➤ *To confirm if the current setup supports VXLAN, run:*

```
ethtool -k $DEV | grep udp_tnl
```

Example:

```
ethtool -k ens1f0 | grep udp_tnl  
tx-udp_tnl-segmentation: on
```

ConnectX-4 family devices support configuring multiple UDP ports for VXLAN offload. Ports can be added to the device by configuring a VXLAN device from the OS command line using the "ip" command.

Note: If you configure multiple UDP ports for offload and exceed the total number of ports supported by hardware, then those additional ports will still function properly, but

will not benefit from any of the stateless offloads.

Example:

```
ip link add vxlan0 type vxlan id 10 group 239.0.0.10 ttl 10 dev ens1f0 dstport 4789
ip addr add 192.168.4.7/24 dev vxlan0
ip link set up vxlan0
```

Note: dstport' parameters are not supported in Ubuntu 14.4.

The VXLAN ports can be removed by deleting the VXLAN interfaces.

Example:

```
ip link delete vxlan0
```

Important Note

VXLAN tunneling adds 50 bytes (14-eth + 20-ip + 8-udp + 8-vxlan) to the VM Ethernet frame. Please verify that either the MTU of the NIC who sends the packets, e.g. the VM virtio-net NIC or the host side veth device or the uplink takes into account the tunneling overhead. Meaning, the MTU of the sending NIC has to be decremented by 50 bytes (e.g 1450 instead of 1500), or the uplink NIC MTU has to be incremented by 50 bytes (e.g 1550 instead of 1500)

© Copyright 2024, NVIDIA. PDF Generated on 06/06/2024