ETHERNET OVER INFINIBAND

Evgenii Smirnov and Mikhail Sennikovsky

ProfitBricks GmbH

April 10, 2018
ETHERNET OVER INFINIBAND: CURRENT SOLUTIONS

- **mlx4_vnic**
  - Currently deprecated
  - Requires specialized HW (BridgeX gateway)

- **VXLAN over IPoIB**
  - Some stability issues in IPoIB on our workload patterns
  - IPoIB in CM mode doesn’t scale well with multi-threaded transfers
  - IPoIB in UD mode has lower performance for single-threaded transfers
  - Extra complexity due to many layers
    - IB/IPoIB/IPv6/UDP/VXLAN/Ethernet
OUR EOIB SOLUTION

- Is a high-speed and scalable Ethernet over InfiniBand linux driver
- Allows up to $5 \times 10^8$ virtual networks separated on the InfiniBand layer
- Presented as a standard Ethernet network interface with all benefits like `ip` tool, `ethtool`, bridging, vlans etc.
- Supports checksum and segmentation offloading on mlx4
- Does not require specific IB hardware (e.g. BridgeX)
- Similar to EoIB concept presented by Ali Ayoub at OFA-2013
- Is an equivalent of Omni-Path VNIC for InfiniBand
Example with three hosts and three separated virtual networks

Infiniband fabric

Network 0xF000:0xC100: Host A, Host B, Host C
Network 0xF050:0xC100: Host A, Host B
Network 0xF000:0xC200: Host B, Host C
MAIN CONCEPTS: VES & VPORT

- Ethernet Overlay Network on top of InfiniBand UD Transport
- Broadcast domain is identified by PKEY + MLID pair
- VES - Virtual Ethernet Switch
  - Can have one or more VPORTs
  - Works as a self-learning switch with its Forwarding Database (FDB)
- VPORT (Virtual Port)
  - Performs actual data transmission
  - Identified by VES and QPN
- Virtual Ethernet interface uses VPORT API to talk to the EoIB network
Despite using the same MLID, VPORT 1 or VPORT2 cannot communicate to VPORT 3, as it uses different PKEY and subscribed to a different MGID.
EoIB uses Mellanox mlx4_vnic encapsulation header format

- Signature and Version set to values used by mlx4_vnic
- Header enables:
  - HW RX path offloads (e.g. checksum validation)
  - “Software TSS” on HCAs older than ConnectX-4 (see above TSS field)
VES works as a self-learning switch with a Forwarding Database (FDB).

FDB maps MAC + VLAN to LID + QPN
- Both 802.1q and 802.1ad (QinQ) are supported.

FDB is updated based on incoming traffic.

If FDB mapping for the destination MAC+VLAN does not exist, the outgoing frame is sent via IB multicast.
EXAMPLE: PING DIAGRAM

- Broadcast ARP for IP that is not in ARP cache
- Learn MAC/VLAN to LID/QPN mapping
- Use the MAC/VLAN to LID/QPN mapping to determine the unicast destination address
- Multicasting ping on Infiniband if there is no FDB entry for destination MAC
- Learn MAC/VLAN to LID/QPN mapping
- Use the MAC/VLAN to LID/QPN mapping to determine the unicast destination address
The following HW offloads are supported (currently only for mlx4):

- IP / TCP / UDP checksum calculation on TX
- IP / TCP / UDP checksum validation on RX
- Large send offload
- Transmit side scaling (TSS)
- Receive side scaling (RSS)
Basic configuration example:

```
# ip li add eoib0 type eoib ves 0xf000:0xc100
# ip li del eoib0
```

Other settings can be specified with `ip link add`:

- Generic settings like ethernet address, number of rx and tx queues, etc.
- EoIB-specific settings like IB device & port, FDB size, IB rate, Queue to MSI-X interrupt mapping, Q_Key

Ethtool configuration support

- tx/rx/tso offloads, statistics etc.
Uperf multithread tcp test results summary

Bandwidth for 1KB packets

Bandwidth for 128KB packets

Measured on Intel Xeon E5-2680 and ConnectX-3 VPI in FDR10 mode
Linux kernel 4.4, Mellanox OFED 3.4
BENCHMARKING: CPU USAGE

Uperf multithread tcp test results summary

Measured on Intel Xeon E5-2680 and ConnectX-3 VPI in FDR10 mode
Linux kernel 4.4, Mellanox OFED 3.4
FUTURE PLANS

 TODOs we plan to work on
  • Support for mlx5
  • Open-source it and offer to the upstream kernel
  • Performance improvements and tuning

 TODOs we do NOT plan to work on (so far ;)
  • Path speed discovery
  • Support of multiple InfiniBand subnets

If you are working on a similar project, we would be happy to cooperate.
THANK YOU

Development team:

Eugene Crosser <evgenii.cherkashin@profitbricks.com>
Evgenii Smirnov <evgenii.smirnov@profitbricks.com>
Mikhail Sennikovsky <mikhail.sennikovskii@profitbricks.com>
Sergii Riabchun <sergii.riabchun@profitbricks.com>

ProfitBricks GmbH, the IaaS-Company: [http://www.profitbricks.com](http://www.profitbricks.com/)