Using New DPDK Port Representor by Switch Application like OVS

RONY EFRAIM, MELLANOX
• What we brought?
• Why we need?
• What it include?
• First open source example.
• Mellanox solution.
What we brought?

Switch representation
Switch representation

- Few years ago we made a change and brought the switchdev to the kernel.
- All the customers wanted in DPDK
- DPDK 18,11 bring the switch management DPDK
- New API for switch representation introduced in DPDK, for switch application like OVS.
- [https://doc.dpdk.org/guides/prog_guide/switch_representation.html](https://doc.dpdk.org/guides/prog_guide/switch_representation.html)
Why we need?

Accelerate Switch And Packet Processing
The need for ASAP²

- **Software Virtual Switches** create performance burden:
  - High CPU utilization
  - Limited Throughput
  - Higher Latency

- **VNF offload**
  - VNF packet processing is CPU bounded
  - Common pipeline processing elements can be done in Hardware
    - e.g. flow classification, ACL drop rules etc.
Smart NIC

- Smart NIC can offload the entire Datapath
- Embedded Switch (eSwitch)
  - Virtual Switch implemented inside the Adapter
  - Flow based switching
  - Overlay tunnel (VxLAN or others) Encap/Decap
- SR-IOV enable direct access from VM to the Adapter
- Control plane and software path run in DPDK
- ASAP² enables SDN on SR-IOV
  - Separation of control and data plane
  - Open vSwitch interfaces to the user remain untouched
    - The hardware offloads are transparent to the user
  - User does not need changes in his environment

Control plane

Data plane
Software based vs Hardware based

**Traditional Model: All Software**
High Latency, Low Bandwidth, CPU Intensive

**ConnectX: Hardware Offload**
Low Latency, High Bandwidth, Efficient CPU

First flow packet

Fallback FRWD path

HW forwarded Packets
What it include?

Port representor
RTE flow API
Port Representors

- Representor ports are an `ethdev` modeling of eSwitch ports
- The VF representor supports the following operations
  - Send packet from the host CPU to VF (OVS Re-injection)
  - Receive of eSwitch “miss” packets
  - Flow configuration (add/remove)
  - Flow statistics read for the purposes of aging and statistics
First open source example.

Open vSwitch
OVS support for HW offload

- Changes are made only in the DPDK datapath code.
- HW offload of flow using rte_flow.
- Packets forwarded by the DPDK datapath are transmitted on the representors and forwarded by the e-switch to the respective VF or to the wire.
Mellanox solution.

Accelerated Switching & Packet Processing
RoCE and DPDK from the VM + Overlay support

- HW switch Acceleration
  - HW overlay support (VXLAN etc)
  - 0% for switch packets 10’s Mpps to VM

- VM/containers Packets Processing (VNF)
  - Use native Mellanox PMD over VF
  - Use RoCE (RDMA)

- Low latency and high Bandwidth to the VM
### OVS over DPDK VS. OVS Offload – ConnectX-5

<table>
<thead>
<tr>
<th>Test</th>
<th>ASAP2 Direct</th>
<th>OVS DPDK</th>
<th>Benefit</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 Flow VXLAN</td>
<td>66M PPS</td>
<td>7.6M PPS</td>
<td>8.6X</td>
</tr>
<tr>
<td>60K flows VXLAN</td>
<td>19.8M PPS</td>
<td>1.9M PPS</td>
<td>10.4X</td>
</tr>
</tbody>
</table>

ConnectX-5 provide significant performance boost Without adding CPU resources
Q & A