Hardware offloads

Past present and future

Oz Shlomo – ozsh@mellanox.com
The challenge of data plane platforms

Flexibility

Performance
Hardware offload performance

OVS Performance - Connect-X 5 100G

- PPS
- Flows: 10, 100, 1K, 10K, 100K, 400K
- VXLAN (SW), VXLAN (HW), VXLAN + CT (SW), VXLAN + CT (HW), VXLAN + CT + NAT (SW), VXLAN + CT + NAT (HW)
HW offloads system integration

- SR-IOV
  - Bare metal performance
- vDPA SW mode - VirtIO SW acceleration
  - No vendor driver on the Guest OS
  - Native live migration support
  - High scale
- vDPA HW mode - VirtIO offload
  - ConnectX 6DX, Bluefield, Bluefield 2
HW offload in OVS

```
  ovs-vsctl set Open_vSwitch . other_config:hw-offload=true
```

- Flow add/delete/stats events are forwarded to HW offload thread
  - Kernel offload using TC
  - DPDK offload using rte_flow
HW offload control plane

- OVS data plane rules are converted to TC filters
  - TC is used to configure Traffic Control in the Linux kernel
  - One component is a packet classifier
  - The flower classifier is a flow based filter
- OVS data plane rules are downloaded to the NIC via rte_flow API

recirc_id(0), in_port(3), eth(src=24:8a:07:a5:28:02, dst=24:8a:07:a5:28:01), eth_type(0x0800) actions:2

tc filter add dev ens1f0_1 ingress protocol ip chain 0 prio 3 flower dst_mac 24:8a:07:a5:28:01 src_mac 24:8a:07:a5:28:02 action mirred egress redirect dev ens1f0_0

flow create 1 ingress transfer pattern eth src is 24:8a:07:a5:28:02 dst 24:8a:07:a5:28:01 type is 0x0800 / / end actions port_id id 0 / end
TC filters are processed **before** openvswitch
- The openvswitch kernel driver hooks to the rx_handler

```bash
tc filter add dev ens1f0_1 ingress protocol ip chain 0
  prio 3 flower
dst_mac 24:8a:07:a5:28:01
src_mac 24:8a:07:a5:28:02
ip_flags nofrag
action mirred egress redirect dev ens1f0_0

recirc_id(0),in_port(3),eth(src=24:8a:07:a5:28:02,dst=24:8a:07:a5:28:01),eth_type(0x0800),ipv4(frag=no) actions:2
```
Netlink netdev – data plane

*With recirculations (e.g. CT)*

- TC filters are processed **before** openvswitch
- The openvswitch kernel driver hooks to the rx_handler

```
recirc_id(0), in_port(4), ct_state(-trk), eth(), eth_type(0x0800), ipv4(proto=6, frag=no),
actions: ct(zone=1), recirc(0x9)
```

tc filter add dev ens1f0_0 ingress prio 1 chain 0 proto ip
flower src_mac 24:8a:07:a5:28:01 ip_flags nofrag ct_state
-trk action ct zone 1 pipe action goto chain 9

```
recirc_id(9), in_port(4), ct_state(+trk+new),
eth(), eth_type(0x0800), ipv4(proto=6, frag=no), actions: ct(zone=1, commit), 2
```
Netlink netdev data plane processing pipeline

- Packet is processed by openvswitch
  - HW offload is disabled
  - OVS/TC limitation
- Packet is processed by TC
  - NIC vendor limitation
- Packet is partially processed by TC
  - Recirculation was partially offloaded
DPDK netdev offload

- Currently only partial offload is supported
  - Rules matches are marked in HW
  - HW marks are associated with netdev filters

- Add full offload support
  - Matches and actions are performed in HW
DPDK Netdev - tunnels offload

- Tunnel encapsulation translates to a single raw_encap action
- Tunnel decapsulation is composed of 2 flows
  - br_phy flow – Classify tunnel (e.g. UDP port match), decap and (implicit) recirc
  - br_int flow – The application flow

- Realize the HW model when offloading tnl_pop action
  - Map tunnel vport to a HW group
  - HW registers (DPDK mark, meta, tags) are required for multi-table state
DPDK netdev connection tracking offload

- Multi table architecture
- Query and restore flow state information using mark, meta and tags

recirc_id(0), in_port(2), ct_state(-trk), eth_type(0x0800), ipv4(proto=6, frag=no), actions: ct(zone=1), recirc(0x5)
recirc_id(0x5), in_port(2), ct_state(+new+trk), eth_type(0x0800), ipv4(proto=6, frag=no), actions: ct(commit, zone=1), 3
recirc_id(0x5), in_port(2), ct_state(+est+trk), eth_type(0x0800), ipv4(proto=6, frag=no), actions: 3
Takeaways

- HW offload is the way to get high performance in OVS
- HW offload supports sriov and virtio
- HW offload will not break system logic - Misses on HW will be handled by software
- HW offload is added incrementally based on SW platform and NIC vendor support
- Kernel datapath HW offload integration uses TC
  - HW model is implemented in the vendor driver
- DPDK datapath HW offload integration uses rte_flow
  - HW model is implemented in OVS
Thank You