Linux Networking & OVS as a Pipeline

Speaker: Dan Daly, Intel

Team: Namrata Limaye, Venkata Suresh Kumar, Sandeep Nagapattinam, Nupur Uttarwar, Intel
Flow vs. Pipeline

Flow

Exact Match
↓
Megaflow Match

Issues:
• Combo Explosions
• Butterfly Effects

Pipeline

Fixed Function
↓
Programmable

Issues:
• A priori knowledge
• Wildcards in Software
P4 Control Plane

https://ipdk.io/p4cp-userguide/

Re-implemented ‘P4 OVS’ as part of Linux Networking as a Pipeline
Golden Rule

Linux Behavior == Linux as a Pipeline Behavior
Differences must be called out / disallowed

Rule:

Linux Kernel is the Golden Model
### Port Representors

Pipeline acts as a co-processor

#### Host Netdevs
I/O to VMs, containers, & Apps
- port0 netdev
- port1 netdev
- bond0 netdev
- vlan100 netdev
- vf1 netdev

#### Host Data Representors
Representors to Netdevs on Host
- \( \leftarrow \) represents host-port0
- \( \leftarrow \) represents host-port1
- \( \leftarrow \) represents host-bond0
- \( \leftarrow \) represents host-vlan100
- \( \leftarrow \) represents host-vf1

#### Network Facing Representors
Representors to Netdevs facing the Network
- uplink0 \( \rightarrow \) Trapped pkts to/from uplink0
- uplink1 \( \rightarrow \) Trapped pkts to/from uplink1
- lag0 \( \rightarrow \) Trapped pkts to/from lag0
- vlan100 \( \rightarrow \) Trapped pkts to/from vlan100
- vf1 \( \rightarrow \) Trapped pkts to/from VF #1

Host Netdevs can be separate from the host.
Example: Map Tunnels → VLAN
Example as seen from OVS

VLAN Maps to Ports/Bonds Based on Host Config

- host-vlan1600
- bridge1600
- vxlan16000

- host-vlan200
- bridge200
- vxlan2000

- host-vlan100
- bridge100
- vxlan1000

- host-port0
- bridge0
- uplink0

- host-port1
- bridge1
- uplink1

- host-bond0
- bridge-lag
- bond0

VXLAN Maps to Ports/Bonds Based on RemoteIP
Example: Map Tunnels \(\rightarrow\) VLAN

Example as seen from OVS

```
$ ovs-vsctl add-br bridge0
$ ovs-vsctl add-port bridge0 host-port0
$ ovs-vsctl add-port bridge0 uplink0
$ ovs-vsctl add-br bridge1
$ ovs-vsctl add-port bridge1 host-port1
$ ovs-vsctl add-port bridge1 uplink1
```

```
$ ovs-vsctl add-br bridgelag
$ ovs-vsctl add-port bridge1 host-bond0
$ ovs-vsctl add-port bridge1 lag0

$ ovs-vsctl add-br bridge100
$ ovs-vsctl add-port bridge100 VXLAN1000 -- set interface VXLAN1000 type=VXLAN
  options:remote_ip=10.100.0.1 options:key=1000
$ ovs-vsctl add-port bridge100 host-bond0 tag=100

$ ovs-vsctl add-br bridge200
$ ovs-vsctl add-port bridge200 VXLAN2000 -- set interface VXLAN2000 type=VXLAN
  options:remote_ip=10.100.0.15 options:key=2000
$ ovs-vsctl add-port bridge200 host-bond0 tag=200
...

$ ovs-vsctl add-br bridge1600
$ ovs-vsctl add-port bridge1600 VXLAN16000 -- set interface VXLAN16000 type=VXLAN
  options:remote_ip=10.100.0.1 ip options:key=16000
$ ovs-vsctl add-port bridge1600 host-bond0 tag=16000
```
Configuration Example

Bond config "create lag" → Linux Create LAG → netlink → ...
... → infrap4d-krnlmon → SAI → TDI → Pipeline Target

OVS config "create tunnel" → Linux Create Tunnel → netlink → ...
... → infrap4d-krnlmon → SAI → TDI → Pipeline Target
Also → Linux Maps ‘RemoteIP’ → MAC using ARP/ND table

OVS "Learns MAC" → P4Runtime Call → ...
... → infrap4d-p4runtime → TDI → Pipeline Target
Virtual L2 Pipeline

**Ingress Classify**
Match On: Ports & VLANs, PFs & VFs, Tunnels & VNIs

**Virtual L2**
vL2 Lookup By: BridgeID, DMAC, SMAC=Learned
Default Action: Trap to Linux Control Plane Port Representors

**Egress Modify**
Modify For: Ports & VLANs, PFs & VFs, Tunnels & VNIs
Make it a Combo!

Ingress Classify
- Match On: Ports & VLANs, PFs & VFs, Tunnels & VNIs
- 

Virtual L2
- vL2 Lookup: BridgeID, DMAC, SMAC=Learned
- Default Action: Trap to Linux Control Plane (Port Representers)
- 

Egress Modify
- Modify For: Ports & VLANs, PFs & VFs, Tunnels & VNIs
- 

Egress Ordering

Parallel Pipelining
Summary

Linux as a Pipeline
• Full Feature Fidelity
• Enables Optimizations & Offloading
• New! Patch to Open vSwitch
  • [https://github.com/ipdk-io/ovs](https://github.com/ipdk-io/ovs)
• Supports Real World Combos →
Thank You!
Virtual L3 Pipeline

Ingress Classify
- Match On: Innermost DestIP & SrcIP

IP Route +
- VRF ID
- Ingress Interface

IPsec
- vL3 Lookup By: VRF ID, DestIP, SourceIP
- Egress Interface

Egress Modify
- Modify For: IP Routing, IPsec Encrypt, IPsec Decrypt