A Brief Guide to Virtual Switching

Franck Baudin (Red Hat)
Billy O’Mahony (Intel)
Enterprise Data Center

- Larger Packet Mix for Endpoint Use
- 10/25G connectivity
- Software switching
- Out-of-box platform software
- SDN controllers Priority
- Mainstream hardware features
- Live Migration Typical

Telco Network Infrastructure

- Smaller Packets in Network Switching
- 10/25G connectivity and greater
- Software augmented with hardware
- Custom platform software
- Network Functions Virtualization
- Low jitter/latency
- Lower Downtime
A Brief History of some Open vSwitches

- **DPDK** 2013
- **Open vSwitch** 2007
- **tungstenfabric** 2018
- **FPGA** 2019+

Intel is leading the innovation with industry partners

*Other names and brands may be claimed as the property of others*
Kernel v. User-space Datapath

Kernel model slow:
- Packet needs to be copied across kernel/user-space boundary in both host and guest
- Kernel code responsible for packet I/O, tunneling, etc.

DPDK model fast:
- NIC DMA directly into user-space.
- DPDK memory allocation with hugepages reduces TLB misses.
- DPDK in host & guest copies directly from host user-space to guest.
## Dataplanes Compared

<table>
<thead>
<tr>
<th>Forwarding Tables</th>
<th>OVS-DPDK</th>
<th>vRouter-dpdk</th>
<th>FD.io vPP</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Openflow tables “NORMAL-ACTION” (L2 LEARNING)</td>
<td>Interface table</td>
<td>Interface: Physical &amp; Virtual, BONDING</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Next-hop table</td>
<td>Layer 2: VLAN Q, Q-in-q</td>
</tr>
<tr>
<td></td>
<td></td>
<td>htable (flow table)</td>
<td>Layer 3: Unicast/MultiCast</td>
</tr>
<tr>
<td></td>
<td></td>
<td>mtries (routing table/FIB)</td>
<td>FIB Lookup</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Bridge table (for each interface)</td>
<td>Layer 4: Sessions/Endpoint</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>TM: Classifiers, POLICING, NAT ...</td>
</tr>
<tr>
<td>Tunnel / Overlay</td>
<td>VXLAN, VLAN, GRE, MPLS, GENEVE, STT</td>
<td>MPLSoGRE/MPLSoUDP, VXLAN ...</td>
<td>IPSEC, SFC, SRv6, GRE, VXLAN, GENEVE, L2TP, PPP, GTPU ...</td>
</tr>
<tr>
<td>Control Plane</td>
<td>Openflow, OVN, Neutron ML2, ...</td>
<td>XMPP (XML), VROUTER agent</td>
<td>VPP-API (LOCAL IPC) with control plan apps for NETCONF/YAnG, ODL, neutron ML2,</td>
</tr>
<tr>
<td></td>
<td></td>
<td>INTEGRATES WITH nEUTRON, ODL, etc.</td>
<td></td>
</tr>
</tbody>
</table>
Dataplanes Compared

Each node is applied to all relevant packets before next node
- Leverages instruction locality cache
- No flow caching – performance scales with #flows
- Graph is easily extensible

Open Flow tables traversal is squashed to single datapath rule:
- Processing an existing flow is fast
- New flows must be processed by Ofproto

Ofproto Classifier
- Open Flow Tables
- Wildcard
- Ordered entries - tuples can overlap
- Has next table.

Datapath Classifier
- Wildcard matches
- Multiple *disjoint* match-tuple
- Flows expire

Exact Match Cache
- Exact match
- 8,192 entries
- N-way cache
OpenVSwitch In Red Hat Portfolio and communities

OVS is distributed by layered products / communities, **not** by the OS distribution.

*Other names and brands may be claimed as the property of others*
Red Hat OpenStack supported SDN/Neutron plugins

Some are based on OVS, some on tungsten fabric, some on FD.io, some on others.

https://access.redhat.com/ecosystem/search/#/category/Software

*Other names and brands may be claimed as the property of others*
**OpenStack/NFV Datapath Performances**

- **Low Range**
  - Kernel OVS
  - No tuning, default deployment
  - Up to 50 Kpps

- **Mid Range**
  - OVS-DPDK
  - Up to 4 Mpps per NUMA socket*
  - Lack of NUMA Awareness
  - [OpenStack Blueprint NUMA aware vswitch (Rocky)](https://example.com)

- **High Range**
  - SR-IOV OVS HW Offload TC-flower w/ SR-IOV**
  - 20+ Mpps per core (line rate?)
  - OVS 2.9, OpenStack Queens
  - RHOSP 13/RHEL 7.5 (Tech Preview)

---

Throughput measured in Packets per second with 64 Byte packet size (VSPerf PVP)
OVS OFFLOAD product roadmap

Kernel offload: RHOSP13 (TP*)

Complete offload: RHOSP15 (TP*)

(*)Technology Preview: https://access.redhat.com/support/offering/techpreview

Two offload already in-progress!

VF specific driver => virtio in HW (vDPA)
Performance measured PVP

- PVP: Physical -> Virtual -> Physical
- Traffic pattern until 2020 at least
- Platform calibration: testpmd PP and PVP
- Supported vSwitches: testpmd, OVS (kernel and DPDK), FD.io

Field reality

- Multi-queue works in lab only
- Realistic configuration (OpenFlow pipeline)
- Real traffic pattern
- All items above are WIP
Production’s path

**End user** focused
Realistic expectations
Troubleshooting materials
Distributions/SDN commercial support

Innovation’s path

**Developers** focused
Supported SDNs / Nth. Bound API
Containers, Source-Routing, ...
10x performances (in some lab...)

“one size fits all” or modular stack allowing multiple vSwitches/vRouters?
Thank you!