Jan Scheurich – Ericsson
Mark Gray – Intel

OvS-DPDK performance optimizations to meet Telco needs
Introduction

- OVS-DPDK in complex NFV environments
- What determines performance in OVS-DPDK?
- OVS 2.5 performance baseline in L3-VPN use case
- Find and address performance bottlenecks
- Achieved improvements in OVS 2.6 and beyond
- Potential future work
What is NFV?
Virtual Network Functions

- Firewall
- Load Balancer
- Deep Packet Inspection
- Content Filter
- Carrier Grade Network Address Translation
- Evolved Packet Gateway

OVS - DPDK
Typical OVS Benchmark Setup

OVS-DPDK

br0

dpdk0  dpdk1

Trivial OpenFlow Pipeline
Typical OVS Configuration for NFV

- **SDN Controller**
- **VNF**
  - **br-int** (OpenFlow)
  - **vxlan0** (User-space (Native) tunneling)
  - **VTEP IP**
  - **bond0**
  - **dpdk0**
  - **dpdk1**
- **br-phy** (Normal mode)
- **Complex OpenFlow pipelines**

**Table 38**
- LFIB Table
- Match mpls label
- Remote NH Group
- Local NH Group
- Internal tunnel/TSTTable 36
- ELAN SMAC Table
- Match
- Miss
- Filter Equal table
- Ericsson service chaining pipeline
- DHCP Table (16)
- Match
- Miss
- DNAT Table (25)
- NAT - FIB Table (28)
- Inbound NAPT Table (44)
- Match
- Subnet Route
- SNAT Table (26)
- Match
- Fib Table 21
- Local NH
- Remote NH
- Table Miss
- Default Route
- Subnet Route
- ARP Table (80)
- Match
- Miss

**VM Port**
- Ext Tunnel of-port
- Int. Tunnel of-port
- VM OF Port
- DHCP Ext Tunnel Table (18)
- Match
- miss
What affects OVS-DPDK performance?

- **Exact Match Cache**
  - Logically, Single Table per datapath thread
  - Exact Match
  - 8192 entries / per thread

- **Datapath Classifier**
  - Logically, Single Table per datapath thread
  - Wildcard Matches
  - 65536 entries
  - Each table is implemented as a priority list of subtables in order to implement wildcards

- **Ofproto Classifier**
  - Logically, Multiple (up to 255) Open Flow tables in pipeline per Open vSwitch bridge
  - Wildcard Matches
  - Each table is implemented as a priority list of subtables in order to implement wildcards

Cost of lookup increasing
What affects OVS-DPDK performance?

**Exact Match Cache**
- Logically, Single Table per datapath thread
- Exact Match
- 8192 entries / per thread

**Datapath Classifier**
- Logically, Single Table per datapath thread
- Wildcard Matches
- 65536 entries
- Each table is implemented as a priority list of subtables in order to implement wildcards

**Ofproto Classifier**
- Logically, Multiple (up to 255) Open Flow tables in pipeline per Open vSwitch bridge
- Wildcard Matches
- Each table is implemented as a priority list of subtables in order to implement wildcards

Cost of lookup increasing

![Diagram](image-url)
What affects OVS-DPDK performance?

**RX Cost**
- Interface Type
- Number of packets in batch

**Lookup Cost**
- Mini flow extract
- Table Type
- Table Configuration
- Flow Type
- Number of flows in each table

**Action Cost**
- Action Type
- Recirculation

**TX Cost**
- Interface Type
- Number of packets in batch
What affects OVS-DPDK performance?

- **Exact Match Cache**
  - Logically, Single Table per datapath thread
  - Exact Match
  - 8192 entries / per thread

- **Datapath Classifier**
  - Logically, Single Table per datapath thread
  - Wildcard Matches
  - 65536 entries
  - Each table is implemented as a priority list of subtables in order to implement wildcards

- **Ofproto Classifier**
  - Logically, Multiple (up to 255) Open Flow tables in pipeline per Open vSwitch bridge
  - Wildcard Matches
  - Each table is implemented as a priority list of subtables in order to implement wildcards

Cost of lookup increasing
What this work focuses on:

- **Exact Match Cache**
  - Logically, Single Table per datapath thread
  - Exact Match
  - 8192 entries / per thread

- **Datapath Classifier**
  - Logically, Single Table per datapath thread
  - Wildcard Matches
  - 65536 entries
  - Each table is implemented as a priority list of subtables in order to implement wildcards

- **Ofproto Classifier**
  - Logically, Multiple (up to 255) Open Flow tables in pipeline per Open vSwitch bridge
  - Wildcard Matches
  - Each table is implemented as a priority list of subtables in order to implement wildcards

Cost of lookup increasing

**EXECUTE ACTION**

**Lookup cost**

**Recirc cost**

**Tx cost**

**EXECUTE ACTION**

**miss** in red arrow

**miss** in red arrow
Ericsson Benchmark: Performance Baseline: OVS 2.5.0

L3-VPN over VXLAN Throughput (single core, 64 byte)

- **OVS 2.5.0**
- **Ericsson HiPvS**
- **VPP 16.06**

**source:** Ericsson

**CPU:** Single socket, Xeon CPU E5-2658 v2 @ 2.40GHz, 10 cores + HT, 640K L1, 2560K L2, 25MB L3 cache
**NIC:** Intel 82599, 2 x 10Gigabit/s, Memory: 4 banks of 16GB DDR3 1600 MHz
Cost Breakdown of L3-VPN in OVS 2.5
(4000 L4 flows)

- RX: 40.0%
- Flow extraction: 6.8%
- EMC Lookup: 12.5%
- Megaflow Lookup: 1.7%
- Action: 6.9%
- TX: 8.2%
- Other: 23.9%

```
pmd thread numa_id 0 core_id 1:
  emc hits:1512270
  megaflow hits:1732461
  miss:0
  lost:0
  polling cycles:138949317 (5.78%)
  processing cycles:2263790775 (94.22%)
  avg cycles per packet: 740.51
  avg processing cycles per packet: 697.68
```
Optimization Activities (1/2)

- Replace tuple space classifier with a trie based classifier
- Faster crc32 hash function
- TX packet batching
- Data structure alignment
Optimization Activities (2/2)

- dpcls per in_port with sorted subtables
- Probabilistic EMC insertion
- More meaningful PMD performance debug info
- Combine actions for TX to tunnel to avoid recirculation

OVS 2.6
Ericsson Benchmark: OVS Performance Improvements

L3-VPN over VXLAN Throughput (single core, 64 byte)

- **OVS 2.5**
- **VPP 16.09**
- **OVS 2.6**
- **OVS 2.6 + patches**

**source:** Ericsson

**CPU:** Single socket, Xeon CPU E5-2658 v2 @ 2.40GHz, 10 cores + HT, 640K L1, 2560K L2, 25MB L3 cache

**NIC:** Intel 82599, 2 x 10Gigabit/s, **Memory:** 4 banks of 16GB DDR3 1600 MHz
Cost Breakdown after Optimizations (4000 L4 flows)

source: `perf top`
Future Efforts

- Lookup key on demand
- Action cost reduction
- Others?
**Summary**

- OVS-DPDK is being deployed as a virtual switch in complex NFV environments
- Exposes OVS to more complex configurations and traffic profiles than in traditional use cases
- Targeted optimization and redesign efforts have successfully improved the performance of OVS-DPDK for a typical NFV use case by a factor of 2.6
- Collaboration between teams with different experiences and viewpoints can yield great results!
Disclaimers

Intel does not control or audit third-party benchmark data or the web sites referenced in this document. You should visit the referenced web site and confirm whether referenced data are accurate.

Intel technologies’ features and benefits depend on system configuration and may require enabled hardware, software or service activation. Performance varies depending on system configuration. **No computer system can be absolutely secure.** Check with your system manufacturer or retailer or learn more at [intel.com].
Questions?
References

- DPCLS per in_port with sorted subtables
  commit 3453b4d62a98f1c276a89ad560d4212b752c7468

- Data structure alignment
  [link](http://openvswitch.org/pipermail/dev/2016-October/080654.html)

- Probabilistic EMC insertion
  [link](http://openvswitch.org/pipermail/dev/2016-November/xxxxx/html)

- PMD performance debug info
  [link](http://openvswitch.org/pipermail/dev/2016-November/xxxxx/html)

- TX Batching
  [link](http://openvswitch.org/pipermail/dev/2016-November/xxxxx/html)

- TX to tunnel ports without recirculation (combine actions)
  [link](http://openvswitch.org/pipermail/dev/2016-November/xxxxx/html)