Integrating and using DPDK with Open vSwitch

Kevin Traynor <ktraynor@redhat.com>
Aaron Conole <aconole@redhat.com>
DPDK Summit - San Jose – 2017

#DPDKSummit
● Open vSwitch dates from 2009
  ○ First commit by Ben Pfaff
  ○ Date: Wed Jul 8 13:19:16 2009 -0700
  ○ Import from old repository commit 61ef2b42a9c4

● DPDK first integrated into Open vSwitch in 2014
  ○ First commit by Gerald Rogers and Pravin Shelar
  ○ Date: Mon Mar 24 19:23:08 2014 -0700
  ○ dpif-netdev: Add DPDK netdev.

● 10x performance improvement for small packets

● Challenge that Open vSwitch was not built for DPDK

#DPDKSummit
Overview of Challenges

- DPDK is greedy
- DPDK wants to use its own data structures for everything
- Everything gets done at initialization
- Inconsistencies between PMDs
- Debugging practically non-existent
- Long-term support issues
- OvS creates its own threads for control and datapath functionality
- It does not use DPDK slave lcores
- By default one of the OvS control threads is used for DPDK init
- Keeps OvS userspace control thread model and adds threads dynamically for datapath
- Not necessary to stick to DPDK threading model
● OvS has its own concept of a packet in userspace dp_packet

● Potentially that could have been an issue but...dp_packet was implemented in a layered manner

● This allows for build time option to back ovs dp_packets with rte_mbufs
● Different accessor functions are used depending on the backing

● One of the few places where a #define DPDK is needed
● DPDK initialization (rte_eal_init) requires arguments

● DPDK init arguments passed to the ovs-vswitchd as cmd line params
  ○ ovs-vswitchd --dpdk -c 0x8 -n 4 --socket-mem 1024,0 ...

● Changed to optional OVSDB parameters with defaults
  ○ ovs-vsctl set Open_vSwitch . other_config:dpdk-lcore-mask=0x8
  ○ ovs-vsctl set Open_vSwitch . other_config:dpdk-mem-channels=4
  ○ ovs-vsctl set Open_vSwitch . other_config:dpdk-socket-mem=1024,0

● Defaults allow less user knowledge and “normal” ovs-vswitchd cmd line
● OVSDB allows for dynamic initialization of DPDK
  ○ ovs-vsctl set Open_vSwitch . other_config:dpdk-init=true

#DPDKSummit
● Devices bound through dpdk-devbind.py and later driverctl

● Devices had to bound before ovs-vswitchd started

● vswitch user had to select the DPDK port
  ○ ovs-vsctl add-port br0 <dpdk>0 -- set Interface dpdk0 type=dpdk

● Changed to use arbitrary name and PCI address/vdev name
  ○ ovs-vsctl add-port br0 myportname -- set Interface myportname type=dpdk
    options:dpdk-devargs=0000:01:00.0

● Some device ports cannot be specified by PCI, so need a more generic usable way to specify them
- PMD's are used for I/O with Hardware

- New NIC's can have some integration issues
  - e.g. Seg fault OVS
  - e.g. Differences in how number of reported Rx queues used

- Don't assume 0 integration effort because it works with testpmd!
Really difficult to debug when things go wrong with DPDK side of OvS
  - Very few tools available for debugging - when things go wrong, where to look?
  - Sparse logs, many require recompile to enable, and usually aren’t useful
  - Application needs to actively enable debugging related features
  - Some failures impact parts of the system that seem unrelated (nature of async processing, and work queues)

If it’s difficult for developers, imagine how it is for users.

Tuning requires specialized knowledge, and little documentation is available upstream.
● DPDK LTS - Used where possible - Yuanhan++ / Luca++
Upgrades

- API/ABI (Where to start!)
  - Preventing dynamic linking - means that 2 versions of DPDK need to carried
  - One standalone package, and one integrated with OVS
  - OVS developers very clued in to DPDK, but will not be same with other apps

- Been a known integration pain point since the beginning (which is one of the reasons why the OvS uses a light shim)
Kevin Traynor <ktraynor@redhat.com>
Aaron Conole <aconole@redhat.com>