Developing CI/CD for NICs
DPDK PMD using DTS

RAMI ROSEN (DECEASED)
SHAHAR BELKAR – TOGA NETWORKS

- A friend
- A colleague
- A mentor
- A Linux Kernel expert, the author of "Linux Kernel Networking", Apress, 2014
- https://ramirose.wixsite.com/ramirosen
Agenda

- DTS
  - Overview and Background
  - Basic Setup
  - Reports
  - UNH IOL (Community Lab)
  - Performance Test Lab Dashboard
  - Advantages

- Challenges enabling DTS on a new PMD
- DTS as part of CI/CD
- DTS as a validation environment
- Status & Roadmap
DTS: Overview and background

- **DTS - DPDK Test Suite**
  - [https://doc.dpdk.org/dts/gsg/](https://doc.dpdk.org/dts/gsg/)
  - [https://doc.dpdk.org/dts/test_plans/](https://doc.dpdk.org/dts/test_plans/)

- **DTS was started in 2014 by Intel**
  - A Framework written in Python 2.7
  - Consists of *functional tests* and *benchmarking tests*
  - There are over 140 test modules (Called Test Suites in DTS semantics)
  - Uses SW packet generators: scapy, dpdk-pktgen and TRex (by Cisco)
  - Also HW packet generator, IXIA (Using it's TCL API)

- **Supports testing of various NICs from:**
  - Intel  -  Mellanox  -  Cavium  -  Huawei

- **Generates reports in the following forms:**
  - RST (ReStructuredText)  -  Excel  -  PDF
DTS: Basic Setup

- Typically, we use two hosts, connected directly by a cable; one is the tester, the second one is the DUT (Device Under Test)
  - You need to set SSH keys so that you can SSH from Tester to DUT without prompting for a password
  - Alternatively, connect an IXIA HW traffic generator instead of the tester

- The DUT has DPDK installed on it, by default under /root/dpdk
This document provides test plan for benchmarking of Rss reta (Redirection table) updating a Poll Mode Driver (PMD) in userland runtime configurations. The content of Rss Redirection table are not defined following reset of the Memory Configuration registers. System software must initialize the table prior to enabling multiple receive queues. It can also update the redirection table during run time. Such updates of the table are not synchronized with the arrival time of received packets.

### Redirection table Tests

<table>
<thead>
<tr>
<th>packet index</th>
<th>hash value</th>
<th>hash index</th>
<th>queue id</th>
<th>actual queue id</th>
<th>pass</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0x827a87fa</td>
<td>250</td>
<td>1</td>
<td>1</td>
<td>true</td>
</tr>
<tr>
<td>1</td>
<td>0xeb0c4370</td>
<td>112</td>
<td>0</td>
<td>0</td>
<td>true</td>
</tr>
<tr>
<td>2</td>
<td>0x2190f621</td>
<td>33</td>
<td>1</td>
<td>1</td>
<td>true</td>
</tr>
<tr>
<td>3</td>
<td>0x52ff77d2</td>
<td>210</td>
<td>1</td>
<td>1</td>
<td>true</td>
</tr>
<tr>
<td>4</td>
<td>0x152b9464</td>
<td>100</td>
<td>0</td>
<td>0</td>
<td>true</td>
</tr>
<tr>
<td>5</td>
<td>0x7c5d50ee</td>
<td>238</td>
<td>1</td>
<td>1</td>
<td>true</td>
</tr>
<tr>
<td>6</td>
<td>0x7065cecc</td>
<td>204</td>
<td>0</td>
<td>0</td>
<td>true</td>
</tr>
</tbody>
</table>
DTS: UNH IOL (Community Lab)

- DTS is used by UNH IOL
  - University of New Hampshire Inter-Operability Lab
  - DPDK Performance Test Lab
  - https://www.iol.unh.edu/testing/hpc/dpdk
  - https://lab.dpdk.org/results/dashboard/

- For each patch which is sent over the dpdk-dev mailing list, an automated performance regression testing is launched

- The degradation/boost in performance (in percentage), relative to a known baseline, is published on the public site of the lab

- Currently this is done on a variety of Intel & Mellanox NICs

*Taken from DPDK website - http://core.dpdk.org/lab/
DTS: Performance Test Lab Dashboard

- Dashboard

<table>
<thead>
<tr>
<th>Status</th>
<th>Patch</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pending</td>
<td>54416 v2</td>
</tr>
<tr>
<td></td>
<td>[19,08, v2]</td>
</tr>
<tr>
<td>Waiting</td>
<td>54415 v2</td>
</tr>
<tr>
<td></td>
<td>[v2] my release...</td>
</tr>
<tr>
<td>Waiting</td>
<td>54412-54411 v3</td>
</tr>
<tr>
<td></td>
<td>my fix...</td>
</tr>
<tr>
<td>Pass</td>
<td>54404-54407 v3</td>
</tr>
<tr>
<td></td>
<td>virtio: my fix...</td>
</tr>
</tbody>
</table>

- Performance report

<table>
<thead>
<tr>
<th>Result</th>
<th>frame_size (bytes)</th>
<th>txd/rxd (descriptors)</th>
<th>throughput Difference (Mpps)</th>
</tr>
</thead>
<tbody>
<tr>
<td>PASS</td>
<td>64</td>
<td>256</td>
<td>0.68600</td>
</tr>
<tr>
<td>PASS</td>
<td>128</td>
<td>256</td>
<td>0.26300</td>
</tr>
<tr>
<td>PASS</td>
<td>256</td>
<td>256</td>
<td>-0.05100</td>
</tr>
<tr>
<td>PASS</td>
<td>512</td>
<td>256</td>
<td>0.03100</td>
</tr>
<tr>
<td>PASS</td>
<td>1024</td>
<td>256</td>
<td>0.00400</td>
</tr>
<tr>
<td>PASS</td>
<td>1518</td>
<td>256</td>
<td>-0.02000</td>
</tr>
</tbody>
</table>
DTS: Advantages

- Doesn’t require to re-invent the wheel
- Open source (share knowledge with community)
- Updated before each DPDK formal release
- Rich documentation and user guides
- Easy to use and friendly environment
DTS: Advantages – Cont.

- Adopted by the major vendors
- Compatible with different NICs and platforms
- Performance measurements
- Supports virtualization (VM configuration, different hypervisors supported etc)
- Aligns the commits base line and improves the quality of open source patch
- Enables a standardized and fully automated environment for CI/CD
Challenges Enabling DTS on a New PMD (IN200)

- Some of the tests are specific to intel NICs (i40e, FVL, Niantic, etc)
  - E.g. the ddp_test (Dynamic Device Personalization), is Intel FVL specific
  - In some cases, the test module name includes it

- Some of the tests required adaptations in the DTS code
  - checksum_offload
  - I2fw
  - tso

- Some tests required adaptation in the PMD
  - We found out that in order for the jumbo frame test to work, we need to set DEV_RX_OFFLOAD_JUMBO_FRAME | DEV_RX_OFFLOAD_SCATTER for RX, and DEV_TX_OFFLOAD_MULTI_SEGS for TX in the hinic_dev_infos_get() method
  - Vhost PMD Xstats test required specific values and naming
DTS as part of CI/CD

**Why?**
- Reduce issues which can occur when submitting new patches
- Submit higher quality patches and avoid later fix patches
- Track any performance degradation in advance

**How?**
- Update the DPDK repo on a daily basis against DPDK master. Make sure there are no new DTS failures
- Run a small portion of DTS for acceptance per developer commit to the local repo
- Execute DTS as part of the framework, so that it will run every night and prior to each submission to the mailing list
DTS as a Validation Environment (SW/FW/HW)

- Robust environment
- Set a base line for higher quality PMDs
- Allows full flexibility
- Assist each other in creating the base line, fixing bugs and generalizing test cases

Requires changes in the environment:
- Clean up after failure/crash (catch and clean)
- Prevent parallel executions
- Create separation for specific vendor feature (e.g. DDP)
- Unified test framework (already proposed by Intel)
  - Task assignment
  - Resource management
  - Case management
  - Reports
  - Guest agent
Status & Roadmap

- Triggering a few DTS tests after each commit (basic acceptance)
- Triggering the DTS before each commit to the mailing list
  - Many TestSuites are passing, where some required adaptations due to the specific NIC’s behavior
  - Committed the patches to the mailing list, targeting DPDK 19.08, after relevant DTS TestSuites passed

- Future work
  - Create a smart acceptance per commit (with time boundaries – up to 10 min)
  - Automatically deploy the environment to any matching setup (vs. dedicated)
  - Widen the coverage (OS types and platforms)
  - Automatically run full DTS test suite on a nightly basis
  - Encourage the DPDK community to run DTS as part of their CI/CD
  - Upgrade to Python 3, as Python 2.7 is end of life in half a year (January 1, 2020)
Questions?

Shahar Belkar
Backup