TRex Ate My Virtual Routers - Adventures in Building a Network Lab

Mark Brochu
Internet2
Background

- Historically, Internet2 has had a single lab located at the GlobalNOC, Indiana University.
- This lab had served network and software engineers well.
- As time progressed, the lab became utilized more and more for testing of software features critical to the network (OESS).
- It became clear that for trial testing/dev work, it would be ideal to have an environment that was a bit less rigid.
- We used Juniper’s Junosphere product, and, it did the job, kinda….
  - Limited to the types of hardware and JunOS versions
  - Limited number of Resource Units assigned. $$
  - Booting up large environments sometimes got wonky… VMX routers would boot up without vFPCs…
  - Weird issues on occasion with interconnectivity
  - Connecting Junosphere lab to other networks outside of Junosphere was wonky
  - Support for the product seemed to be limited to 2 guys
Current Environment

New Lab Setup

Internet2 Backbone

MERI/T/internet2 Ann Arbor

Indiana University

I2 Dev Lab

IU TESTLAB
High Level Overview - Dev Lab

Wait - something’s missing…..

Oh yeah, a traffic generator.

Why? Because engineers like to test as much as possible.

- Testing Router ACLs
- Testing QoS
- Testing Load Balancing (Hashing) Algorithms
- Testing Netflow Collector Features (regression testing)

What we have had -

- IXIA Optlxia XM2 chassis and Xcellon-Lava Load Module - Good for Link Qualification - Not stateful
- Xena Load Tester - At IU - Would work, but limited to 10G.
Enter Cisco TRex (https://trex-tgn.cisco.com/)

TRex
Realistic traffic generator

TRex is an open source, low cost, stateful and stateless traffic generator fuelled by DPDK. It generates L4-7 traffic based on pre-processing and smart replay of real traffic templates. TRex amplifies both client and server side traffic and can scale up to 200Gb/sec with one UCS.

TRex Stateless functionality includes support for multiple streams, the ability to change any packet field and provides per stream statistics, latency and jitter.

New Advanced Stateful functionality includes support for emulating L7 traffic with fully-featured scalable TCP layer.
Intel DPDK

“Data Plane Development Kit (DPDK) greatly boosts packet processing performance and throughput, allowing more time for data plane applications. DPDK can improve packet processing performance by up to ten times. DPDK software running on current generation Intel® Xeon® Processor E5-2658 v4, achieves 233 Gbps (347 Mpps) of L3 forwarding at 64-byte packet sizes. As a result, telecom and network equipment manufacturers (TEMs and NEMs) can lower development costs, use fewer tools and support teams, and get to market faster.” - Intel

- Set of data plane libraries and network interface controller drivers for fast packet processing
- Created by Intel in 2010 - Moved to Linux foundation 2017
- Userspace application, which bypasses the kernel networking stack, direct access to Network Hardware
- Various optimizations such as hugebuffers, cache alignment, others, well beyond scope of this talk
Installing TRex

- For many DPDK apps, typically need to download and compile the SDK
  https://github.com/DPDK/dpdk
- Once SDK is installed, compile your DPDK app with the target set to the DPDK installation directory
- TRex is much easier.
  - Follow these directions - https://trex-tgn.cisco.com/trex/doc/trex_appendix_napatech.html
  - Install latest Napatech Drivers (https://www.napatech.com/downloads/)
  - Make some adjustments to the Napatech Settings
  - Start Driver
  - Download and compile latest version of TRex with support for Napatech
    - $path_to_install_dir/linux_dpdk/.b configure --with-ntacc
    - $path_to_install_dir/linux_dpdk/.b build
  - opt/trex/dpdk_setup_ports.py -i (creates a YAML config file that TRex uses to figure out your network ports, CPU settings, etc)
100G TReX / Napatech Proof of Concept

Juniper QFX5200-32C
100G-SR4

Napatech NT200A01

Dell R640
Xeon E5-2630@2.40GHz, 2x8 Cores
96GB RAM, 4x10G NIC

~ $10,000
The Results?

Global Statistics

<table>
<thead>
<tr>
<th>connection</th>
<th>total_tx_L2 : 62.71 Gb/sec</th>
</tr>
</thead>
<tbody>
<tr>
<td>version</td>
<td>total_tx_L1 : 82.3 Gb/sec</td>
</tr>
<tr>
<td>cpu_util.</td>
<td>total_rx : 62.71 Gb/sec</td>
</tr>
<tr>
<td>rx_cpu_util.</td>
<td>total_pps : 122.47 Mpkt/sec</td>
</tr>
<tr>
<td>async_util.</td>
<td>drop_rate : 0 b/sec</td>
</tr>
<tr>
<td></td>
<td>queue_full : 0 pkts</td>
</tr>
</tbody>
</table>

Port Statistics

<table>
<thead>
<tr>
<th>port</th>
<th>0</th>
<th>1</th>
<th>total</th>
</tr>
</thead>
<tbody>
<tr>
<td>owner</td>
<td>root</td>
<td>root</td>
<td></td>
</tr>
<tr>
<td>link</td>
<td>UP</td>
<td>UP</td>
<td></td>
</tr>
<tr>
<td>state</td>
<td>TRANSMITTING</td>
<td>IDLE</td>
<td></td>
</tr>
<tr>
<td>speed</td>
<td>100 Gb/s</td>
<td>100 Gb/s</td>
<td></td>
</tr>
<tr>
<td>CPU util.</td>
<td>97.89%</td>
<td>0.0%</td>
<td></td>
</tr>
<tr>
<td>Tx bps L2</td>
<td>62.71 Gbps</td>
<td>0 bps</td>
<td>62.71 Gbps</td>
</tr>
<tr>
<td>Tx bps L1</td>
<td>82.3 Gbps</td>
<td>0 bps</td>
<td>82.3 Gbps</td>
</tr>
<tr>
<td>Tx pps</td>
<td>122.47 Mpps</td>
<td>0 pps</td>
<td>122.47 Mpps</td>
</tr>
<tr>
<td>Line Util.</td>
<td>82.3 %</td>
<td>0 %</td>
<td>82.3 %</td>
</tr>
</tbody>
</table>

| Rx bps   | 177.23 bps | 62.71 Gbps | 62.71 Gbps |
| Rx pps   | 0.06 pps   | 122.47 Mpps | 122.47 Mpps |

opackets | 11479793510 | 0 | 11479793510 |
ipackets | 5 | 11483587182 | 11483587187 |
obytes  | 872475339904 | 0 | 872475339904 |
lbytes  | 1790 | 872650680382 | 872650680382 |
tx-pkts | 11.48 Gpkts | 0 pkts | 11.48 Gpkts |
rx-pkts | 5 pkts | 11.48 Gpkts | 11.48 Gpkts |
tx-bytes | 872.48 GB | 0 B | 872.48 GB |
rx-bytes | 1.79 KB | 872.65 GB | 872.65 GB |

errors | 0 | 0 | 0 |
errors | 0 | 0 | 0 |

status: \nPress 'ESC' for navigation panel... status: [OK]

tui>start -f stl/bench.py -m 100% --port 0 --t vm=cached
Great Job!
Still more to do:

- Investigate Stateful support
- Determine if we’re using optimal configuration
- Test elephant flows
- Introduce impairments (latency)

Special Thanks to:

Karl Newell, Internet2
The folks at Napatech
Jeff Hagley and the TSG staff at Internet2