Measuring Geneve Tunnel Throughput for Hardware Accelerated DataPath with OVN!!

Pradipta Sahoo  
Senior Software Engineer

Haresh Khandelwal  
Principal Software Engineer
Agenda

● To provide an operator's outlook on Geneve hw-offload overlay network performance with ml2/OVN core service provided by OpenStack.
● Explore Trex software traffic-generator to simulate encapsulated traffic with Geneve protocol.
● To identify the challenges hardware offload network while using Geneve hw offload test with OVN.
● Measure the line rate throughput numbers in Perf scenario
● Share the observation / findings
Possible Test Methodology

1. **Virtual Trex Traffic generator on Tenant Layer:**
   - Configured trafficgen emulator in VM running in the same tenant network.
   - During the test, we observed virtual trafficgen provide up to 6-7 Mpps due lack of hardware acceleration.

2. **Bare Metal Trex node with Intermediate Gateway host:**
   - Gateway do the encap/ dcap packet between Trafficgen and Tenant VM and expected to take some cycle to process the packets.
   - No changes required in existing trafficgen code.
   - Since the OVN gateway is software, it can not sustain line rate from traffic generator.

3. **Bare Metal Trex host without Intermediate Gateway Host:**
   - Using existing Trex’s library to compose encapsulated packet with GENEVE header.
   - Transmit the encapsulated packets with linerate without additional processing through TEP gateway host.
   - Able to perform the L2-test.
   - Traffic generator established on outside of OpenStack cluster.
   - Ability to to compose packets for multiple vms.
Geneve Protocol Requirements

  - Outer MAC Header (Source, Destination)
  - Outer VLAN Header (Dot1Q)
  - Outer IP Header (Source, Destination)
  - UDP Protocol (Source, Destination)
  - GENEVE Header (option_length, protocol, VNI, options_class, options_type, options_len, options_data)
  - Internal MAC Header
  - Internal IP Header
  - Internal Protocol
  - Payload

```
Ether(src='40:a6:b7:0b:e9:b1',dst='fa:16:3e:40:4d:7d')/
Dot1Q(vlan=177,type='IPv4')/
IP(proto=17,src='172.17.2.163',dst='172.17.2.60')/
UDP(sport=57825,dport=6081)/
GENEVE(optionlen=2,proto=0x6558,vni=6,options_class=0x0102,options_type=0x80,options_len=1,options_data=b"\x00\x03\x00\x02")/
Ether(src='fa:16:3e:7f:a1:49',dst='fa:16:3e:a0:1f:28',type=0x0800)/
IP(src='192.168.2.1',dst='192.168.2.181')/
UDP(dport=1000,sport=2000)/pad
```
Software Specifications

1. Red Hat OpenStack Platform release 16.2.0 Beta (Train)
   ○ OS: Red Hat Enterprise Linux release 8.4 (4.18.0-305.el8.x86_64)
   ○ OpenvSwitch:
     ■ rhosp-network-scripts-openvswitch-2.15-4.el8ost.1.noarch
     ■ openvswitch2.15-2.15.0-15.el8fdp.x86_64
     ■ rhosp-openvswitch-2.15-4.el8ost.1.noarch
     ■ openvswitch-selinux-extra-policy-1.0-28.el8fdp.noarch
     ■ Network-scripts-openvswitch2.15-2.15.0-15.el8fdp.x86_64
   ○ OVN:
     ■ rhosp-ovn-2021-4.el8ost.1.noarch
     ■ ovn-2021-host-21.03.0-21.el8fdp.x86_64
     ■ ovn-2021-21.03.0-21.el8fdp.x86_64
     ■ Rhosp-ovn-host-2021-4.el8ost.1.noarch

2. Trex
   ○ v2.89: https://trex-tgn.cisco.com/trex/release/v2.89.tar.gz
   ○ OS: Red Hat Enterprise Linux release 8.4 (4.18.0-305.el8.x86_64)
Geneve Scapy library status in Trex

- Challenges on existing Geneve scapy library ([v2.4.3](https://external_libs/scapy-2.4.3/scapy/contrib/geneve.py))
  - Geneve options fields not work for OVN class with metadata fields
- Enhanced the Geneve headers as per OVN metadata requirement.
- Raised the PR in upstream community: [https://github.com/secdev/scapy/pull/3329](https://github.com/secdev/scapy/pull/3329)
- Build the Traffic STL profile GENEVE scapy library and build Scapy packet with Geneve headers.

```python
>>> packet.show2()
###[ GENEVE ]###
version = 0
optionlen = 2
oam = 0
critical = 0
reserved = 0
proto = TEB
vni = 0x4
reserved2 = 0x0
\options \ 
|###[ Geneve Options ]### |
| classid = Open Virtual Networking (OVN) |
| type = 0x80 |
| reserved = 0 |
| length = 1 |
| data = '\x00\x04\x00\x05' |
```
Challenges and Solution

- Initial Challenges:
  - OVN controller discarded packets with any of below reason:
    - Incorrect outer headers with VLAN.
    - Incorrect Protocol
    - Incorrect Geneve metadata information.
    - Incorrect inner packet headers

- Solution
  - Trex Server:
    - Outer header Src MAC and IPs can be same subnet of OpenStack Tenant network.
    - Outer header Dst MAC and IPs belongs OpenStack Compute (DUT) tenant network interface with same Tenant VLAN id.
    - Traffic profile update with correct metadata fields.
  - OVN Controller:
    - To recognize outside Geneve packets respective interface of Trex host need to register as a OVN SouthBound Chassis.
    - Map logical port of each tenant network to respective chassis and collect the tunel_key for source and destination traffic.
  - No changes need in OpenStack Compute (DUT) and OpenStack services.
  - TestPMD VM can be run as default IO mode.
Configuration topology

Trex

- Ip: 172.17.2.160
  - Gw: 172.17.2.60

- Ip: 172.17.2.161
  - Gw: 172.17.2.60

- Ip: 172.17.2.162
  - Gw: 172.17.2.60

- Ip: 172.17.2.163
  - Gw: 172.17.2.60

OVN-SB

- Trex-Port0 Chassis
  - Geneve-Encap
  - 172.17.2.160

- Trex-Port1 Chassis
  - Geneve-Encap
  - 172.17.2.161

- Trex-Port2 Chassis
  - Geneve-Encap
  - 172.17.2.162

- Trex-Port3 Chassis
  - Geneve-Encap
  - 172.17.2.163

OVN-Controller

Compute Chassis
- Geneve-Encap
- 172.17.2.60

OpenStack Compute(DUT)

- mlx5ex
- Tenant VLAN (172.17.2.60)
Logical View of Tenant Network Mapping

- **Tenant Network 1** (VNI=1)
- **Tenant Network 2** (VNI=2)
- **Tenant Network 3** (VNI=3)
- **Tenant Network 4** (VNI=4)

- **OVN-SB**
  - Trex-Port0 Chassis (Geneve-Encap 172.17.2.160)
  - Trex-Port1 Chassis (Geneve-Encap 172.17.2.161)
  - Trex-Port2 Chassis (Geneve-Encap 172.17.2.162)
  - Trex-Port3 Chassis (Geneve-Encap 172.17.2.163)
  - Compute Chassis (Geneve-Encap 172.17.2.60)

- **TestPMD VM1**
  - Tunnel_key: 2

- **TestPMD VM2**
  - Tunnel_key: 2
Geneve End-to-End Packet Journey using Trex

11:32:51.353434 40:a6:b7:0b:e9:b0 > 0c:42:a1:d1:da:98, ethertype 802.1Q (0x8100), length 96: vlan 304, p 0, ethertype IPv4, (tos 0x0, ttl 64, id 1, offset 0, flags [none], proto UDP (17), length 78)

13:29:51.353434 40:a6:b7:0b:e9:b0 > 0c:42:a1:d1:da:98, ethertype 802.1Q (0x8100), length 96: vlan 304, p 0, ethertype IPv4, (tos 0x0, ttl 64, id 1, offset 0, flags [none], proto UDP (17), length 78)

13:29:51.353434 40:a6:b7:0b:e9:b0 > 0c:42:a1:d1:da:98, ethertype 802.1Q (0x8100), length 96: vlan 304, p 0, ethertype IPv4, (tos 0x0, ttl 64, id 1, offset 0, flags [none], proto UDP (17), length 78)

13:46:36.746972 172.17.2.61 > 172.17.2.138, ethertype IPv4 (0x0800), length 34: IP7

13:46:36.746972 172.17.2.61 > 172.17.2.138, ethertype IPv4 (0x0800), length 34: IP7

13:46:36.746972 172.17.2.61 > 172.17.2.138, ethertype IPv4 (0x0800), length 34: IP7

13:46:36.746972 172.17.2.61 > 172.17.2.138, ethertype IPv4 (0x0800), length 34: IP7

Geneve Bi-Directional Traffic Flow using Tenant Network
Performance Test results

<table>
<thead>
<tr>
<th>Frame Size (Bytes)</th>
<th>Duration (min)</th>
<th>Traffic Mode</th>
<th>Traffic</th>
<th>Total_Tx_L1 (Gbps)</th>
<th>Total_Rx (Gbps)</th>
<th>Total_Tx_Rates (Mpps)</th>
<th>Total_Rx_Rate (Mpps)</th>
<th>Drop (PPS)</th>
<th>Cpu_Util (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>128</td>
<td>10</td>
<td>PVP</td>
<td>Bi-Direction</td>
<td>31.02</td>
<td>21.91</td>
<td>42.25</td>
<td>42.25</td>
<td>0</td>
<td>8.12</td>
</tr>
<tr>
<td>256</td>
<td>10</td>
<td>PVP</td>
<td>Bi-Direction</td>
<td>31.25</td>
<td>22.67</td>
<td>22.67</td>
<td>22.67</td>
<td>0</td>
<td>4.37</td>
</tr>
<tr>
<td>512</td>
<td>10</td>
<td>PVP</td>
<td>Bi-Direction</td>
<td>31.11</td>
<td>11.77</td>
<td>11.77</td>
<td>11.77</td>
<td>0</td>
<td>2.24</td>
</tr>
<tr>
<td>1024</td>
<td>10</td>
<td>PVP</td>
<td>Bi-Direction</td>
<td>31</td>
<td>5.99</td>
<td>5.99</td>
<td>5.99</td>
<td>0</td>
<td>1.02</td>
</tr>
<tr>
<td>1500</td>
<td>10</td>
<td>PVP</td>
<td>Bi-Direction</td>
<td>31.06</td>
<td>4.12</td>
<td>4.12</td>
<td>4.12</td>
<td>0</td>
<td>4.67</td>
</tr>
<tr>
<td>9000</td>
<td>10</td>
<td>PVP</td>
<td>Bi-Direction</td>
<td>31.01</td>
<td>0.69307</td>
<td>0.69307</td>
<td>0.69307</td>
<td>0</td>
<td>0.17</td>
</tr>
</tbody>
</table>

![Tx v/s Rx Chart](chart.png)
Conclusion

- Achieved linerate utilisation without intermediate host (ovn gw).
- Suffice the throughput requirement with traffic profiles.
- No changes required in OpenStack Compute and TestPMD. Can be used with others Infrastructure i.e Openshift.
- Very minimal operator level changes in ovn-controller suffice the tunnel requirement.

Next Goal:
  - Integration with Trafficgen to run with binary-search support.
Thank you

Red Hat is the world’s leading provider of enterprise open source software solutions. Award-winning support, training, and consulting services make Red Hat a trusted adviser to the Fortune 500.

linkedin.com/company/red-hat
youtube.com/user/RedHatVideos
facebook.com/redhatinc
twitter.com/RedHat