PoC for L2VPN in&inter DC with EVPN
About EVPN (Ethernet VPN)

- Basic specification is defined on RFC7432 (FEB 2015)
- Expected for functionality
  - Solution for scalability of logical path number limitation
  - Optimization for distribution of multicast traffic
  - Quick convergence
  - Traffic control

Control Plane

- MP-BGP

Data Plane

- MPLS
- PBB
- VXLAN
Features of EVPN

- Features of EVPN
  Reduce ARP flooding by advertising IP/MAC information on Control-Plane
  Support multi homing (Active/Active) in default
  Vender-specific specification (MC-LAG, Stacking, Virtual Chassis)
  Flexible definition for Control-Plane (ex: Separation per ESI)

- Advertising Route type
  Type 1: Ethernet Auto-Discovery (A-D) route
    Per ESI, Multi homing
  Type 2: MAC/IP Advertisement route
    MAC/IP learning for end host
  Type 3: Inclusive Multicast Ethernet Tag route
    Path configuration of BUM traffic, per VLAN or ESI
  Type 4: Ethernet Segment route
    Factor to decide route of multi homing

Reference: Advertisement format of Type 2

<table>
<thead>
<tr>
<th>MAC/IP Advertisement Route</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Route Distinguisher (RD)</td>
<td>(8 octets)</td>
</tr>
<tr>
<td>Ethernet Segment Identifier</td>
<td>(10 octets)</td>
</tr>
<tr>
<td>Ethernet Tag ID</td>
<td>(4 octets)</td>
</tr>
<tr>
<td>MAC Address Length</td>
<td>(1 octet)</td>
</tr>
<tr>
<td>MAC Address</td>
<td>(6 octets)</td>
</tr>
<tr>
<td>IP Address Length</td>
<td>(1 octet)</td>
</tr>
<tr>
<td>IP Address</td>
<td>(0, 4, or 16 octets)</td>
</tr>
<tr>
<td>MPLS Label1</td>
<td>(3 octets)</td>
</tr>
<tr>
<td>MPLS Label2</td>
<td>(0 or 3 octets)</td>
</tr>
</tbody>
</table>
EVPN Network (MAC Learning)

- PE1 and PE4 learn IP/MAC by extended advertisement of BGP
- PE2 and PE3 learn as well

Provider Edge devices
Customer Edge devices (ex: host, router, or switch)
EVPN instance (EVI)
Theme

- It’s difficult to prepare infrastructure to verify network functionality, because hardware appliance is necessary to do it
- Therefore, software based configuration and verification were defined as theme of this project
Environment

- **GoBGP**
  1. BGP Protocol written in Go Language
  2. Supported EVPN
  3. Interoperability with multi vendor in Interop 2016
     Data Plane was implemented with experimental VXLAN version: GoBGP version 1.15, GoPlane (Latest)

- Ixia(Instrument)

- OS : ubuntu16.04
**Configuration 1 (Simplified)**

- Confirmation items
  - Advertisement of IP/MAC
  - Segment separation with vni
  - Segment separation with etag

![Diagram]

- **backbone router**
- **Edge router**

**EVPN instance (EVI)**
- vni 10 etag 10 or 20
- vni 20 etag 10 or 20
Exchange MAC table
goplane log

```
{"Etag":20,"Topic":"VirtualNetwork","level":"debug","msg":"modFdb new path, prefix: [type:macadv][rd:65000:20][esi:single-homed][etag:20][mac:52:54:00:10:75:db][ip:\u003cnil\u003e][labels:[10]], nexthop: 10.1.249.11, withdraw: false","time":"2017-02-02T08:34:32+09:00"}
```

packet capture log

```
08:34:32.543656 IP 10.1.249.11.bgp > 10.1.249.13.39897: Flags [P.], seq 20:123, ack 19, win 227, options [nop,nop,TS val 52171091 ecr 52173163], length 103: BGP
08:34:32.544211 IP 10.1.249.13.39897 > 10.1.249.11.bgp: Flags [.], ack 123, win 229, options [nop,nop,TS val 52175058 ecr 52171091], length 0
```

MAC learning for vtep

- fdb of host3 on communication test from host1 to host3

```
c2:66:a8:a1:c1:b8 dev vtep10 master br10 permanent
fe:54:00:f1:40:92 dev vtep10 dst 192.168.253.11 self permanent

fe:54:00:f1:40:92 dev vtep10 dst 192.168.253.11 self permanent
52:54:00:10:75:db dev vtep10 dst 192.168.253.11 self permanent
```
**Confirmation Result 2 (Simplified)**

- **Separation per vni and etag**

  **Common in etag = 10**

<table>
<thead>
<tr>
<th>from / to</th>
<th>host 3(vni=10)</th>
<th>host 4(vni=20)</th>
</tr>
</thead>
<tbody>
<tr>
<td>host 1(vni=10)</td>
<td>OK</td>
<td>NG</td>
</tr>
<tr>
<td>host 2(vni=20)</td>
<td>NG</td>
<td>OK</td>
</tr>
</tbody>
</table>

  **Common in vni = 10**

<table>
<thead>
<tr>
<th>from / to</th>
<th>host 3(etag=10)</th>
<th>host 4(etag=20)</th>
</tr>
</thead>
<tbody>
<tr>
<td>host 1(etag=10)</td>
<td>OK</td>
<td>NG</td>
</tr>
<tr>
<td>host 2(etag=20)</td>
<td>NG</td>
<td>OK</td>
</tr>
</tbody>
</table>
Configuration2 (Virtual DC)

Throughput measurement between each hosts with ixia
# Confirmation Result 3 (Virtual DC)

## host1 to host3

<table>
<thead>
<tr>
<th></th>
<th>64 byte</th>
<th>64 – 1518 byte</th>
<th>1028 – 1518 byte</th>
<th>1028 – 1518 byte</th>
</tr>
</thead>
<tbody>
<tr>
<td>Throughput</td>
<td>5.545Mbps</td>
<td>83.168Mbps</td>
<td>88.616Mbps</td>
<td>134.46Mbps (Max rate)</td>
</tr>
<tr>
<td>Loss ratio</td>
<td>91.972% loss</td>
<td>5.294% loss</td>
<td>0.001% loss</td>
<td>62.21% loss</td>
</tr>
</tbody>
</table>

## host1 to host2

<table>
<thead>
<tr>
<th></th>
<th>64 byte</th>
<th>64 – 1518 byte</th>
<th>1028 – 1518 byte</th>
</tr>
</thead>
<tbody>
<tr>
<td>Throughput</td>
<td>6.32Mbps</td>
<td>86.183Mbps</td>
<td>91.320Mbps</td>
</tr>
<tr>
<td>Error ratio</td>
<td>89.261% loss</td>
<td>4.824% loss</td>
<td>0.001% loss</td>
</tr>
</tbody>
</table>
What is realized currently

- IP/MAC learning on EVPN-VXLAN was confirmed to work
- Unnecessary ARP Packet was reduced by advertising function of BGP, and it may apply for Traffic balancing function
- Segment separation with VNI and Etag is available
- There was no affection for throughput across EVPN
- Route Redundancy more than 2 routes was not supported, because routes learned by BGP protocol were not configured to Forwarding Table recursively
Summary

- EVPN based on OSS was confirmed in this project
- EVPN with GoBGP and GoPlane was confirmed to work stably in static network configuration of L2VPN
- EVPN may be expected to realize more efficient traffic forwarding like IP/MAC distribution just confirmed in the project