Hewlett Packard
Enterprise

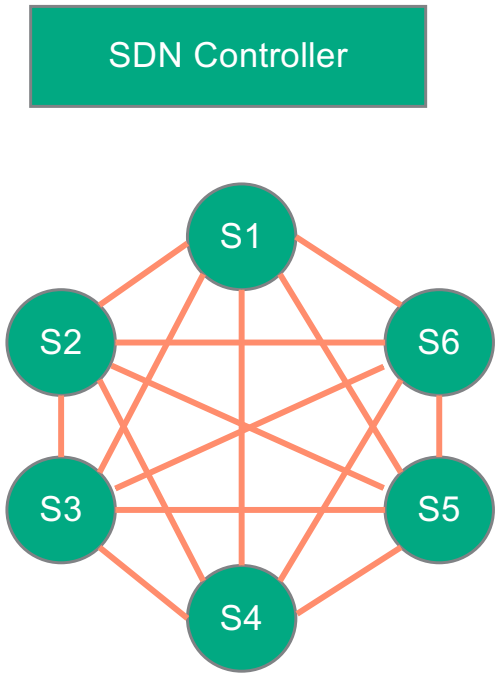
Extend Tunnel Specification with L2 Information

OVSCON 2021 Fall Conference

Vasu Dasari, Master Engineer, HPE/Plexxi

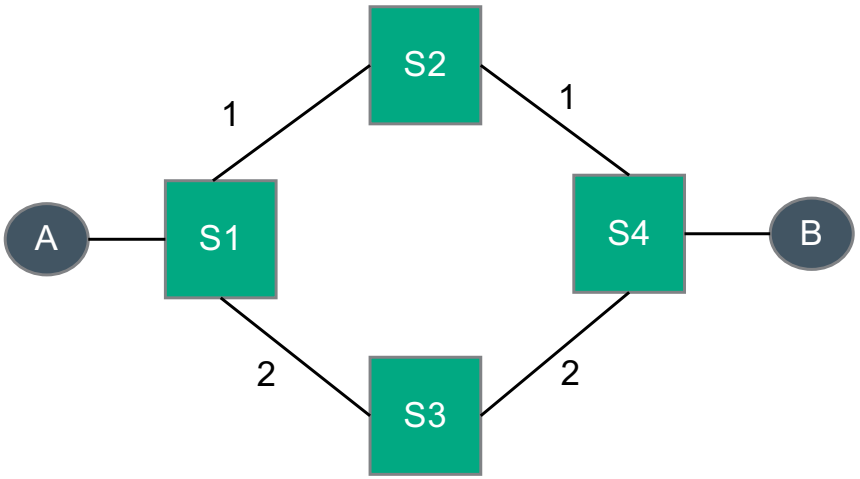
Brief Background

- Yet another SDN Architecture
- Manage conforming programmable switches
  - Open vSwitch (Uses OVSDB, OpenFlow)
  - Plexxi Switches (Uses REST API)
  - Any other conforming switches
- Provide L2/L3 paths to all end devices



Switch Requirements and Limitation

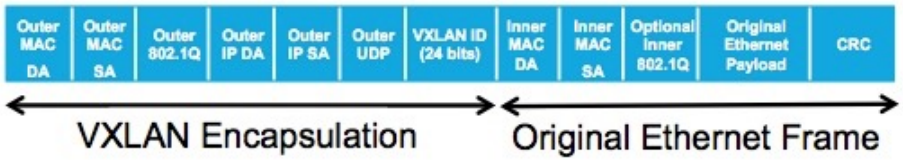
- Ability to spawn VxLAN tunnels
- Ability to specify full VxLAN encapsulation information
  - Both L2 and L3 information
- Ability to specify which outgoing port to take for a tunnel
- Should not rely on native IP stack for VxLAN packet forwarding



OVS Tunnel Specification

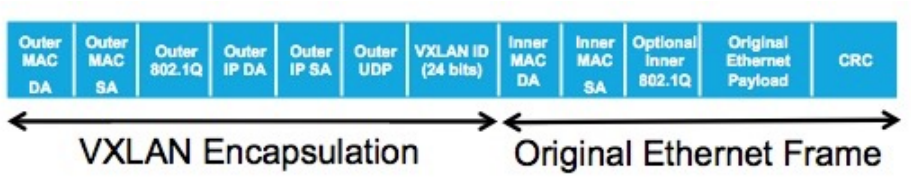
– OVS Tunnel is specified with following parameters
  – Tunnel Type
    – VxLAN/Geneve/GRE, etc
  – Source IP Address
  – Destination IP Address
  – Tunnel Identifier
    – VxLAN Identifier
    – Geneve Identifier

– L2 Information and tunnel's egress port is derived from native IP stack



Proposed Tunnel extensions

- Proposed L2 Extensions
  - Source MAC Address
  - Destination MAC Address
  - Vlan Id
  - Outgoing Port
- For a tunnel to be a fully-specified-tunnel, user must to specify Source Mac
  - Remaining fields can be derived from flow
- Extensions proposed are optional
- First phase of implementation was done in OVS userspace
- Kernel mode can be supported in second phase
- Support all tunnel types supported by OVS
  - L2 Extensions are tunnel-type agnostic



CLI: Tunnel Creation Command

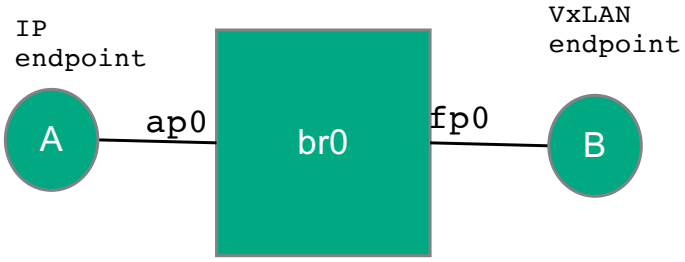
– Changes to tunnel creation command

```
$ ovs-vsctl add-port br0 vxlan_1 --        \
        set int vxlan_1 type=vxlan         \
        options:remote_ip=10.1.1.100       \
        options:local_ip=10.1.1.1          \
        options:key=1000                   \
        options:dst_mac=00:00:0a:01:01:64  \
        options:src_mac=00:00:0a:01:01:01  \
        options:dl_port=fp0                \
        options:vlan_id=100
```

– Specify source-mac alone

```
$ ovs-vsctl add-port br0 vxlan_2 --        \
        set int vxlan_1 type=vxlan         \
        options:remote_ip=10.1.1.100       \
        options:local_ip=10.1.1.1          \
        options:key=1000                   \
        options:dst_mac=flow               \
        options:src_mac=00:00:0a:01:01:01  \
        options:dl_port=flow               \
        options:vlan_id=flow
```

Simple Switch Configuration




```
# Create a tunnel
$ ovs-vsctl add-port br0 vxlan_1 --       \
        set int vxlan_1 type=vxlan        \
        options:remote_ip=10.1.1.100      \
        options:local_ip=10.1.1.1         \
        options:key=1000                  \
        options:dst_mac=00:00:0a:01:01:64 \
        options:src_mac=00:00:0a:01:01:01 \
        options:dl_port=fp0               \
        options:vlan_id=100

# Create a normal flow
$ ovs-ofctl add-flow br0 "actions=normal”
```

Tunnel Metadata extensions

– Following fields will be added to `ovs_tunnel_key_attr`

```
OVS_TUNNEL_KEY_ATTR_DL_PORT,     /* Tunnel datalink port */
OVS_TUNNEL_KEY_ATTR_ETH_SRC,     /* Outer datalink src mac address */
OVS_TUNNEL_KEY_ATTR_ETH_DST,     /* Outer datalink dst mac address */
OVS_TUNNEL_KEY_ATTR_VLAN_ID,     /* Outer datalink vlan_id */
```

– Following fields will be added to `meta-flow.h`

```
MFF_TUN_ETH_SRC     /* tun_eth_src */
MFF_TUN_ETH_DST     /* tun_eth_dst */
MFF_TUN_VLAN_ID     /* tun_vlan_id */
MFF_TUN_DL_PORT     /* tun_dl_port */
```

Flow Configuration

– An example Learn Action

```
ovs-ofctl add-flow br0 'in_port=vxlan_1,
    actions=learn(table=10, NXM_OF_ETH_DST[]=NXM_OF_ETH_SRC[],
            load:NXM_NX_TUN_ID[0..23]->NXM_NX_REG0[0..23],
            load:NXM_NX_TUN_DL_PORT[]->NXM_NX_REG1[0..31],
            load:NXM_NX_TUN_VLAN_ID[0..11]->NXM_NX_REG2[0..11],
            load:NXM_NX_TUN_ETH_SRC[0..31]->NXM_NX_REG3[0..31],
            load:NXM_NX_TUN_ETH_SRC[32..47]->NXM_NX_REG4[0..15],
            load:NXM_NX_TUN_ETH_DST[0..31]->NXM_NX_REG5[0..31],
            load:NXM_NX_TUN_ETH_DST[32..47]->NXM_NX_REG6[0..15],
            )'
```

– Override tunnel parameters using `set_field` option

```
$ ovs-ofctl add-flow br0 'in_port=ap0,
    actions=set_tunnel:1000,
        set_field:00:00:11:11:11:11->tun_eth_dst,
        set_field:100->tun_vlan_id,
        set_field:4->tun_dl_port, vxlan_2'
```

Testing framework

- Added following tests to system-userspace-packet-aware.at
  - 1. datapath - ping over fully specified vxlan tunnel
    - Basic test to verify the functionality
  - 2. datapath - ping over fully specified vxlan tunnel with vlan
    - Same as 1 but VxLAN tunnel is transported over Vlans
  - 3. datapath - ping over fully specified vxlan tunnel all-in-one
    - Setup with tunnels of type 1 and 2 and regular tunnels.
    - This verifies coexistence of all tunnel types
- All tests are written for VxLANs
  - But GRE and Geneve tunnels should also work

Summary

– Implementation on github
  – https://github.com/vasu-dasari/ovs/tree/fst
– Code changes span across 15 files
– Looking forward to collaborate to get the code reviewed
– Any comments?