Content

• Key Concepts
• MPLS/BGP VPN Approach
• VxLAN-GPE NSH Approach
• Open Stack Projects related to SFC / MPLS VPN / BGP
• Open Daylight Projects related to SFC / MPLS VPN / BGP
• OPNFV Projects related to SFC / MPLS VPN / BGP
• Key Takeaways
• References
Key Concepts

- **Classification**
  - Policy-based function to identify / select / match traffic flow with a specific service function chain
  - Customer / network / service specific policies

- **Service Function Chain (aka Service Chain)**
  - An ordered set of service functions and ordering constraints that must be applied to packets and/or frames selected as a result of Classification
  - As simple as a linear chain; or as complex as a service graph with multiple branches

- **Service Function Forwarding Function (VRF or SFF)**
  - Forward traffic to one or more connected SF(s)
  - Transport traffic to another VRF/SFF or classifier
  - Terminate SFC
MPLS/BGP VPN Approach (1 of 2)

- L3VPN as the overlay encapsulation tunnel for routing and traffic flow over SFC topology
  - VM(s) attached to L3VPN
- Controller manages SFC topology, instantiation of SFC, VRF creation and configuration, and route installation
- Support use of existing protocols and PE devices with current capabilities
  - BGP is used for route advertising
  - NETCONG/YANG or XMPP can be used for controller to create and configure VRFs, set up RTs and install routes into service instance interfaces
- Support both physical and virtual deployment
- Multiple Control Plan Protocol (e.g. L3VPN, EVPN) and Multiple Data Plane Encapsulation (MPLS/GRE, VxLAN etc.) supported
### Notes and Description

<table>
<thead>
<tr>
<th>Notes</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Controller</td>
<td>Manage instantiation of SFCs by (1) building a model of the desired topology (SFs, # of instances, connectivity); (2) instantiating of SF instances; (3) calculating routes and instantiating VRFs that will form virtual networks between SF instances; and (4) installing routes to cause traffic to flow into and between SF instances.</td>
</tr>
<tr>
<td>NETCONF-YANG / XMPP</td>
<td>Controller uses NETCONF-YANG and XMPP to create and configure VRFs, set up RTs and install routes into service instance interfaces.</td>
</tr>
<tr>
<td>BGP</td>
<td>Controllers implements RR. Routers uses BGP RR to advertise routes, and interacts with Controller for updates.</td>
</tr>
<tr>
<td>Load Balancing</td>
<td>Combined I-VRF/E-VRF LB and Forward/Reverse Flow LB (for stateful SF) is supported so that each SF in SFC can be separately scaled.</td>
</tr>
</tbody>
</table>
NSH Approach (1 of 4)

• Focus on virtualized SF deployment

• Encapsulation is based on NSH, and tunneling is based on VxLAN-GPE or GRE or Ethernet
  – VM is attached to OVS (L2), and assumes appropriate setup available
  – Multiple tunneling protocols can be applied

• Flow-based classification allows for flexible classification criteria
  – Classifier is required
**Control Plane Functions**

- **Steering Traffic into SFC:**
  - Classifier
- **Traffic Flow through SFC:**
  - Traffic from the network that satisfies classification criteria is encapsulated and directed into an SFP
  - SFF delivers packets to SFs based on SFC Encap
  - Metadata may be added and passed between nodes
  - Transit routers/switches forward based on outer encapsulation

---

**Notes Description**

<table>
<thead>
<tr>
<th>Notes</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Control Plane Function</td>
<td>Manage instantiation of SFCs by (1) domain-wide view of available SF resources; (2) use policies to construct SFCs and associated SFPs; (3) select specific SFs for requested SFCs; (4) provides SFC dataplane info to other components, e.g. SFF; (5) provides metadata and usage info for Classifier; (6) provide info including policy info for other SFC elements to properly interpret metadata</td>
</tr>
<tr>
<td>Service Classifier</td>
<td>Determine what traffic needs to be chained based on policy</td>
</tr>
<tr>
<td>SF Forwarder</td>
<td>Deliver packets / frames to SFs based on info in SFC Encapsulation, e.g. an overlay switch like OVS</td>
</tr>
<tr>
<td>SFC Encapsulation</td>
<td>Carry explicit information used to identify SFP; also enable metadata and context information. It is transport independent</td>
</tr>
</tbody>
</table>
NSH Approach (3 of 4)

- **SF Proxy**
  - Supports SFC-unaware SFs, e.g. legacy SFs
  - Removes and inserts SFC encapsulation on behalf of an SFC-unaware service function

---

<table>
<thead>
<tr>
<th>Component</th>
<th>Insert NSH</th>
<th>Remove NSH</th>
<th>Select SFP</th>
<th>Decrement Service Index</th>
<th>Update Context Header</th>
<th>Service Policy Selection</th>
</tr>
</thead>
<tbody>
<tr>
<td>Classifier</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
<td>✓</td>
</tr>
<tr>
<td>SF Forwarder</td>
<td></td>
<td>✓</td>
<td>✓</td>
<td></td>
<td></td>
<td>✓</td>
</tr>
<tr>
<td>Service Function</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>SF Proxy</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
**NSH Approach (4 of 4)**

- **Base HDR** – Info about service header and payload protocol
  - O: OAM packets
  - C: critical metadata TLV present. MD Type 2 only
  - Next Protocol: protocol type of original packets (IPv4, IPv6, Ethernet)

- **Service Path HDR** – path id and location
  - Service Index: 255 by classifier, and decrement after SF has processed packets
  - Control plane may set different initial value

- **Context HDRs** – Opaque metadata
  - TLV Class: the scope of Type field, e.g. a specific vendor, or specific SDO-allocated
  - Type: specific type of information being carried within the scope of given TLV class
  - Combined TLV.C: 0-127 for non-critical, and 128-255 for critical options
## BGP VPN v.s. NSH

<table>
<thead>
<tr>
<th>MPLS / BGP VPN</th>
<th>NSH</th>
</tr>
</thead>
<tbody>
<tr>
<td>Separate Control Plane, Service Plane and Data Plane</td>
<td>Separate Control Plane, Service Plane and Data Plane</td>
</tr>
<tr>
<td>Classifier not required</td>
<td>Classifier required</td>
</tr>
<tr>
<td>L3VPN as overlay encapsulation tunnel</td>
<td>Encapsulation based on NSH, and overlay tunnel based on VxLAN or GRE or Ethernet</td>
</tr>
<tr>
<td>Both physical and virtual deployment</td>
<td>Virtual deployment</td>
</tr>
<tr>
<td>VM attached to L3</td>
<td>VM attached to L2 (OVS)</td>
</tr>
<tr>
<td>Project Name</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------------------------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
</tbody>
</table>
| **Neutron MPLS VPN-as-a-Service**                | • A Neutron extension that introduces MPLS VPN feature set.  
• APIs for creating, deleting, listing, showing and updating VPN services of a tenant  
• APIs for creating, deleting, listing, showing and updating MPLS Access VPNConnection  
• APIs for creating, configuring, deleting and showing MPLS access connection                                                                                                                                                                                                 |
| **Neutron APIs for Service Chaining**            | • “Port Chain” concept which is an ordered list of Neutron ports that defines the chain  
• “Flow Classifier” concept which specifies what traffic flow enters the service chain  
• **Blueprint** is proposed to address common SFC API independent of backend implementation  
  • Northbound Intent Based Service Chaining API and Intent Engine  
  • Neutron API Extension for Service Chaining  
  • Common Service Chaining Driver API  
  • Service Registration API  
• Driven by the need of OPNFV’s [OpenStack-based VNF Forwarding Graph](#) project
<table>
<thead>
<tr>
<th>Project Name</th>
<th>Description</th>
</tr>
</thead>
</table>
| **BGP-MPLS VPN Extension for OpenStack Networking** | • Provides an API and Framework to interconnect BGP/MPLS VPNs to Openstack Neutron networks, routers and ports.  
• Allow attachment of Neutron networks and/or routers to carrier provided WANs using standard protocols of BGP and MPLS.  
• A vendor neutral API and data model are provided to allow multiple back-ends that can be "plugged in“  
• Support both L3VPN and EVPN |
### Open Daylight Projects related to SFC / MPLS VPN / BGP

<table>
<thead>
<tr>
<th>Project Name</th>
<th>Description</th>
</tr>
</thead>
</table>
| **Group-based Policy (GBP)**      | • A policy framework and engine for ODL  
• Used as Classifier in the setup of ODL SFC and OPNFV SFC (based on VxLAN-GPE NSH)                                                                                                               |
| **Service Function Chaining**     | • Provides the infrastructure (chaining logic, APIs) needed for ODL to provision a service chain in the network  
• Based on VxLAN-GPE NSH approach                                                                                                               |
| **VPN Service**                   | • Implement the infrastructure services required to support L3 VPN service  
• Implementation of L2 VPN and L3 VPN services for data center tenants using technologies like BGP-MPLS VPN and EVPN  
• Build L3 VPN Services using L3VPN based on BGP-MPLS ([RFC 4364](https://tools.ietf.org/html/rfc4364)) in the first Phase (targeted for Lithium).  
• L2 VPN Service based on EVPN (draft-ietf-l2vpn-evpn) is planned for a future ODL release.                                                        |
## OPNFV Projects related to SFC / MPLS VPN / BGP

<table>
<thead>
<tr>
<th>Project Name</th>
<th>Description</th>
</tr>
</thead>
</table>
| **OpenStack-based VNF Forwarding Graph**    | • Integrate OpenStack SFC related components in order to demonstrate an OpenStack based and OpenFlow compliant solution which will dynamically set up VNFFG  
• Deliverables include vendor-neutral SFC requirement and Interface specification, and development of components such as VNFFG Manager, SDN Controller, VNFFG Classifier |
| **Service Function Chaining**               | • Provides the infrastructure to install the upstream ODL SFC implementation project in an NFV environment so as to allow ODL to create SFCs across OPNFV VNFs  
• VxLAN-GPE NSH approach  
• Dependent on ODL GBP and SFC projects |
| **SDN Distributed Routing and VPN**         | • Address integration and deployment of VIM and Virtual networking components to provide Layer 3 VPN services in the OPNFV platform.  
• In collaboration with the related BGPVPN project in OpenStack and supported implementations in SDN controllers (e.g. VPN in ODL) |
Key Takeaways

- Diversity is healthy, but not fragmentation
- For end user, more interests are:
  - Common APIs that can leverage diversified backend implementations
  - A method can support inter-domain/end-to-end SFC use cases across heterogeneous networks
  - A deployment that can leverage existing network capabilities to minimize TCO
References

- **Service Function Chaining (SFC) Architecture**, IETF RFC 7665
- **Service Chaining using Virtual Networks with BGP VPNs**, IETF Internet-Draft
- **Network Service Header**, IETF Internet-Draft
- **Generic Protocol Extension for VXLAN**, IETF Internet-Draft
- **Generic Routing Extension**, IETF RFC 2784 and 2890