A Universal Protocol Mechanism for Network Function Virtualization and Application-Centric Traffic Steering

CSCN'15 Presentation
Changcheng Huang¹, Jiafeng Zhu²

1 – Dept. of Systems and Computer Engineering, Carleton University, Ottawa, Canada
2 – R&D Center, Huawei Technologies Inc., Santa Clara, US
E-mail: huang@sce.carleton.ca¹, jiafeng.zhu@huawei.com²
Outline

- Motivation
- SDN and Issues
- Existing Solutions
- Proposed Solution
- Use Cases
- Conclusions
Outline

- Motivation
- SDN and Issues
- Existing Solutions
- Proposed Solution
- Use Cases
- Conclusions
Motivation

- New services and applications are the driver for future Internet
- Network Virtualization
  - A service by network provider
  - Multiple virtual networks share a substrate network
  - Need a way to identify different virtual networks
- Network Function Virtualization (NFV)
  - Generalization of Service Chaining
  - Built upon virtual networks
    - A recursive service relationship
  - Need a way to identify service chains
- Application-centric Traffic Steering
  - Group-based application (server replication, mobility, etc.)
  - Service providers want to treat user traffic flows differently
  - Switches need a tag to act on
- Recursive service
  - A common service structure in all industries
  - Need identifiers for services at all embedded layers
Motivation (cont’s)

- An example of recursive virtual network
Outline

- Motivation
- SDN and Issues
- Existing Solutions
- Proposed Solution
- Use Cases
- Conclusions
SDN

- SDN earmarked for future Internet
- More flexibility by separating control and data plane
- OpenFlow adopted for communication between control and data
- Allow forwarding based on arbitrary header fields
SDN: OpenFlow

### Switching

<table>
<thead>
<tr>
<th>Switch Port</th>
<th>MAC src</th>
<th>MAC dst</th>
<th>Eth type</th>
<th>VLAN ID</th>
<th>IP Src</th>
<th>IP Dst</th>
<th>IP Prot</th>
<th>TCP sport</th>
<th>TCP dport</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>*</td>
<td>*</td>
<td>00:1f:...</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>port6</td>
</tr>
</tbody>
</table>

### Routing

<table>
<thead>
<tr>
<th>Switch Port</th>
<th>MAC src</th>
<th>MAC dst</th>
<th>Eth type</th>
<th>VLAN ID</th>
<th>IP Src</th>
<th>IP Dst</th>
<th>IP Prot</th>
<th>TCP sport</th>
<th>TCP dport</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>5.6.7.8</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>port6</td>
</tr>
</tbody>
</table>

### Firewall

<table>
<thead>
<tr>
<th>Switch Port</th>
<th>MAC src</th>
<th>MAC dst</th>
<th>Eth type</th>
<th>VLAN ID</th>
<th>IP Src</th>
<th>IP Dst</th>
<th>IP Prot</th>
<th>TCP sport</th>
<th>TCP dport</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>22</td>
<td>drop</td>
</tr>
</tbody>
</table>

OpenFlow/SDN tutorial, Srini Seetharaman, Deutsche Telekom, Silicon Valley Innovation Center
SDN Issues

- SDN limitation
  - Based on existing header fields
  - No field to identify a group-based application
    - Replication servers in data center
    - User mobility and multiple devices
  - No field to identify a service chain
  - No field to identify a virtual network
  - No way to identify services in recursive service structure
SDN Challenges

- Why not VLAN id
  - Limited to a local area network
  - Hard to support recursive services
- Why not IP address
  - Multiple services may share one interface
  - Multiple services may have overlapped address spaces
- Why not port number
  - Traffic with multiple port numbers may share one service
- Combination of L2-4 headers
  - Fragmented flows and bloated flow table
  - Tunneled traffic difficult to identify
Outline

- Motivation
- SDN and Issues
- Existing Solutions
- Proposed Solution
- Use Cases
- Conclusions
Existing Solutions

- New header fields need to be defined
- But where?
- VXLAN
  - Tunneling VLAN over IP
  - Limited to extending VLAN service
- OpenADN
  - Use two new labels: one at Layer 3.5 and one at layer 4.5
  - No end-to-end identifier
  - Complex to implement
  - Do not support recursive service structure
- Serval
  - Add a service access layer between Layer 3 and Layer 4
  - Designed for dynamic binding
  - Hard to traverse a middle box
  - Complex to implement
  - Do not support recursive service structure

2015/10/29
Outline

- Motivation
- SDN and Issues
- Existing Solutions
- **Proposed Solution**
- Use Cases
- Conclusions
Service Forwarding Label (SFL)

- Used to identify a service relationship between a client and a provider at Layer 5
  - Client can be a single user or a group
- ALL SFLs form a service namespace
- SFLs can be stacked to support recursive services
- Administered by network providers
- Unique within the domain of a network provider
- Renewable expiration time
- Added into forwarding entry in Flow Table as a new matching field
Design Consideration

- Why Layer 5
  - Virtual Network service is similar to session layer service
    - Establish, manage and terminate Virtual Network between service provider and network provider
  - Easy access from application
  - Allows non-SDN network traversal
  - Allows middle box traversal

- Why fixed length
  - Easy table match (e.g. a 32 bit label can be 0x5434B8E0)

- Why dynamic
  - Services typically have limited time
  - Allows efficient usage of label space
An example for SFL usage
Outline

- Motivation
- SDN and Issues
- Existing Solutions
- Proposed Solution
- Use Cases
- Conclusions
**Virtual Networks**

- SFL allows wildcard treatment of traffic from a client network and simplifies forwarding
- Support client mobility and group-based applications
- Support overlapped private address spaces
NFV and Service Chaining
Application-centric Traffic Steering

- SFL used to differentiate client traffic with or w/o protection requirement
Migration

Client Network → Legacy IP Network → SDN Network

- Virtual Network 1
- SFL=1

- Virtual Network 2
- SFL=2
Conclusion

- SFL is a universal service identifier
- Used independently or combined with other header fields
- Can identify a group-based application
- Can traverse middle box
- Easy to implement
  - No change to existing socket interface
  - Fit in OpenFlow naturally
References

- https://www.opennetworking.org/
Thank You