OpenFlow

Ihsan Ayyub Qazi

Slides use info from Nick Mckeown
Why Openflow?

• Deployment of new experimental protocols is challenging
  – Huge installed base of protocols and equipment
  – Reluctance by network operators

• Consequence
  – Hard to gain confidence need for deployment
  – Many ideas remain untried and untested
Why Openflow?

• Key desired ingredient
  – Introduce programmability in the network
  – i.e. develop programmable switches/routers for isolating experiments

• Goal
  – Lower barriers for entry of new ideas
  – A researcher/user should be able to an isolated slice of resources across the whole network
    • Slice: portion of network links + packet processing + end-hosts

• GENI is headed in this direction
This paper focuses on campus networks

• Questions
  – How will operators get comfortable with allowing experimental protocols?
  – How will researchers control a portion of their network without disruption others?
  – What functionality is needed to enable experiments?
Some approaches

• Convince equipment vendors
  – Unlikely in short-term
    • Experimental protocols may interfere
    • Barrier-to-entry for new competitors

• Open Software Platforms
  – PC with interfaces
    • Hard to support > 1 Gbps packet processing
      – Although GPUs are changing this (e.g. PacketShader, ACM SIGCOMM 2010)
    • Low port density (need 100+ ports per box)
  – NetFPGAs
    • Limited interfaces (4 in most)
OpenFlow Approach

• How to achieve high performance, support large number of interfaces, achieve traffic isolation, and cater to vendors’ needs for close platforms?

• OpenFlow idea
  – Use the already built-in flow tables (TCAMs)
    • Typically used for firewalls, NAT, QoS, statistics collection
  – Provide an open protocol to program the flow table
    • Allows isolation and control of experimental/production traffic
OpenFlow
OpenFlow: Key Idea
Separate intelligence from datapath

Operators, users, 3rd party developers, researchers, …

New function!
Open Flow

“If header = x, send to port 4”
“If header = y, overwrite header with z, send to ports 5,6”
“If header = ?, send to me”
**OpenFlow: Flow Table**

Exploit the flow table in switches, routers, and chipsets

<table>
<thead>
<tr>
<th>Flow</th>
<th>Rule</th>
<th>Action</th>
<th>Statistics</th>
</tr>
</thead>
<tbody>
<tr>
<td>Flow 1</td>
<td>Rule (exact &amp; wildcard)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Flow 2</td>
<td>Rule (exact &amp; wildcard)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Flow 3</td>
<td>Rule (exact &amp; wildcard)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Flow N</td>
<td>Rule (exact &amp; wildcard)</td>
<td>Default Action</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Statistics</td>
</tr>
</tbody>
</table>
## Flow Table Entry

### OpenFlow Protocol Version 1.0

<table>
<thead>
<tr>
<th>Rule</th>
<th>Action</th>
<th>Stats</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Packet + byte counters</td>
</tr>
</tbody>
</table>

1. Forward packet to port(s)
2. Encapsulate and forward to controller
3. Drop packet
4. Send to normal processing pipeline

+ mask what fields to match

<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>
</tr>
</thead>
</table>

Packet + byte counters
## Examples

### 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>* * 00:1f:.. * * * * * * * port6</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Flow 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>port3       00:2e.. 00:1f.. 0800 vlan1 1.2.3.4 5.6.7.8 4 17264 80 port6</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></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>* * * * * * * * * * * 22 drop</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
## Examples

### 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>

### VLAN

<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>vlan1</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td></td>
<td></td>
<td>port6, port7, port9</td>
</tr>
</tbody>
</table>
OpenFlow Usage

Dedicated OpenFlow Network

Controller

Chip’s code

OpenFlow Protocol

Rule

Action

Statistics

Rule

Action

Statistics

Rule

Action

Statistics

PC
Usage examples

Zartash’s code:
- Static “VLANs”
- His own new routing protocol: unicast, multicast, multipath, load-balancing
- Network access control
- Home network manager
- Mobility manager
- Energy manager
- Packet processor (in controller)
- IPvChip
- Network measurement and visualization
- …
OpenFlow Hardware

- Juniper MX-series
- NEC IP8800
- WiMax (NEC)
- HP Procurve 5400
- Cisco Catalyst 6k
- PC Engines
- Quanta LB4G
Questions

• Why is it that OpenFlow switches can’t do?
  – Arbitrary packet processing
• The first packet of every new flow is sent to the controller
  – What if large number of flows start at the same time?
  – Will controller processing become the bottleneck?
  – How can we address this? Redundant controllers?
• How do recover from Controller failures?
  – Centralized vs. Distributed
  – Internet Philosophy?
• What if OpenFlow were to be deployed throughout the Internet?
  – What changes are needed?
Concludes...

• ...the set of papers on network architectures and design principles

• We looked at the design principles of
  – Internet Architecture
    • Host-to-host communication, fate sharing, thin waist, etc
  – Content Centric Networking (CCN)
    • Content-based naming, routing, and transport
  – OpenFlow
    • An effort to simplify control plan processing and introduce programmability at the network layer
Spectrum of Architectures..

• Internet and CCN: *Two Extreme Design Points*

What about advertisers?, ISPs?

Is this architecture evolvable?

What if usage and traffic patterns change? Would CCN be still be useful?
Thank you!