Software Defined Networking and OpenFlow

Geddings Barrineau
Ryan Izard
What is Software Defined Networking and OpenFlow?
What is SDN?

- Physical separation of network control plane from forwarding/data plane
- Network control
  - Centrally managed
  - Directly programmable
- Network infrastructure
  - Abstracted from applications
How does SDN work?

API Feature A  
API Feature B  
API Feature C  

Network Operating System

Packet Forwarding  
Packet Forwarding  
Packet Forwarding  
Packet Forwarding  
Packet Forwarding

Packet Forwarding
What is OpenFlow?

• Link between SDN control and infrastructure layers

• OF-enabled *infrastructure* communicates with an OF *controller* via the OF *protocol*
How Does OF Work?

- **OpenFlow Controller**
- **Network Services**
- **Custom Services**
- **APIs**

User Applications

Packet Forwarding
User Applications
### OpenFLOW

<table>
<thead>
<tr>
<th>Matches</th>
<th>Actions</th>
</tr>
</thead>
<tbody>
<tr>
<td>• Characteristics of a packet that must be true in order for a flow to apply to the packet</td>
<td>• Applied for any packet that matches the flow</td>
</tr>
<tr>
<td>• Header fields e.g. EtherType, VLAN, src/dst MAC, IP, and transport ports, ingress port</td>
<td>• Tells a packet where to go or can even modify a packet</td>
</tr>
</tbody>
</table>
Basic SDN packet handling example
Basic SDN packet handling example
Basic SDN packet handling example
Why SDN and OpenFlow?

• Traditional networks
  – Static software architecture
  – Closed-source with limited customization
  – Vendor dependence
  – Decentralized control
Why SDN and OpenFlow?

- SDN and OpenFlow
  - Dynamic software architecture
  - Open-source with vast customization
  - Allows large-scale network research
  - Supports integration of custom network services both transparent to and directly controlled by applications
How can Software Defined Networking and OpenFlow be Used?
High-Throughput Data Transfer

- Steroid OpenFlow Service (SOS)
- Enhance TCP connections
  - Large delay-bandwidth product networks
  - TCP windows
- SOS provides
  - Transparent service improvement to user
  - Single and multipath support
High-Throughput Data Transfer

System Architecture Overview

SOS agent
SOS Openflow Controller
SOS pipe
Openflow switch
TCP
Server

Openflow switch
Client
TCP
Mobility over Heterogeneous Networks

- Provide mobility for clients over IPv4
- Entirely SDN and OF-based solution
  - Network-level
    - Migration detection
    - IP address assignment and management
    - Packet routing
  - Client-level
    - Packet routing
    - Interface switching
    - Transparent service to application
Mobility over Heterogeneous Networks

- **Network-Level, Mobile IP:**
  - Home/Foreign Agents OpenFlow Controller
  - Mobile IP tunnels OpenFlow flows
  - DHCP server on OpenFlow controller

- **Client-Level, Vertical Handoff:**
  - Change physical interface ➔ broken socket
  - Open vSwitch + Floodlight OF controller
    - Manage the physical interfaces via SDN
    - End-user sees “always-up” virtual interface
Mobility over Heterogeneous Networks

Client-Level
GENI Cinema

- SDN and OF video streaming service
- Chain of events
  - Client connects to public server and selects video of interest
  - Client receives address of nearby GENI Cinema gateway
  - OF controller (Floodlight) creates path between gateway and appropriate video server
GENI Cinema

- Public Server
- GENI Cinema Gateway
- OpenFlow Controller
- Video Server 1
- Video Server 2
Software Components of an OpenFlow Software Defined Network
Basic SDN Network

Network Operating System

API Feature A

API Feature B

API Feature C

Packet Forwarding

Packet Forwarding

Packet Forwarding

Packet Forwarding
Bottom-Up: An OF-Enabled Switch

1. Power On
2. Bootloader
3. OF-Enabled OS
OF Switch Connection to Controller

- OpenFlow Controller (e.g., Floodlight, NOX, etc.)
- OS (e.g., Linux, Mac, Windows, etc.)
- Hardware

Network Services
Custom Services
APIs

Control Path
OpenFlow

Data Path / Switching Hardware
Flow Table
OF Switch Connection to Controller

- Switch probes for controller
  - Configured with controller IP/port
  - Standalone or secure modes
  - Connection established via TCP/TLS

- Controller learns topology
  - Packet-out LLDP
  - Devices and other participating OF switches discovered and mapped
Application Connection to Controller

- Application Requesting Controller Service(s)
  - Interface to Controller (e.g. REST)
  - OS (e.g. Linux, Mac, Windows, etc.)
  - Hardware

- Network Services
- Custom Services
- APIs
  - OpenFlow Controller (e.g. Floodlight, NOX, etc.)
  - OS (e.g. Linux, Mac, Windows, etc.)
  - Hardware
Application Connection to Controller

- Controller-dependent
- Floodlight provides REST API
  - JSON
  - Modify or query running configuration
  - Expandable with custom modules
- Options are vast with open-source
  - Customize controller APIs and behavior
  - User-application-independent (e.g. a transparent network service like SOS)
The Floodlight OpenFlow Controller

- Open-source OF controller
- Sponsored and supported by Big Switch Networks
- Written in Java and easily used with Eclipse
- Modifiable and expandable to suit any application via modules
- Large developer community and support group
Floodlight Architecture

REST Applications
Applications in any language leveraging services via REST API exposed by controller modules and module applications

REST API

Module Applications
- VNF
- Firewall
- PortDown Reconciliation
- Hub
- Learning Switch

Java API

Floodlight Controller
- Module Manager
- Thread Pool
- Packet Streamer
- Jython Server
- Web UI
- Unit Tests
- Device Manager
- Topology Manager/ Routing
- Link Discovery
- Flow Cache*
- Storage
- Memory
- NoSql*
- Switches
- Controller Memory
- PerfMon
- Trace
- Counter Store

Applications with higher bandwidth communication with controller such as PacketIn’s

Core services of common interest to SDN applications

* Interfaces defined only & not implemented: FlowCache, NoSql
Software Defined Networking and OpenFlow

Geddings Barrineau
Ryan Izard