Software Defined Networking

What is it, how does it work, and what is it good for?

slides stolen from Jennifer Rexford, Nick McKeown, Michael Schapira, Scott Shenker, Teemu Koponen, Yotam Harchol and David Hay
Agenda

• What is **Software Defined Networking (SDN)?**

• What is **OpenFlow?** How does it work?
What is SDN?
The Internet: A Remarkable Story

• Tremendous success
  – from research experiment to global infrastructure

• Enables innovation in applications
  – Web, P2P, VoIP, social networks, virtual worlds

• But, the Internet’s infrastructure remained fairly stagnant for decades
The Internet's Landscape

Applications:

Internet Protocols:

Technologies:

constant innovation

routing, congestion control, naming, …

(TCP/IP, BGP, DNS, OSPF, ECMP, …)

constant innovation
Why Can’t We Innovate?

• Closed equipment
  – software bundled with hardware
  – vendor-specific interfaces

• Over specified
  – slow protocol standardization

• Few people can innovate
  – equipment vendors write the code
  – long delays to introduce new features

Impacts performance, security, reliability, cost...
Networks are Hard to Manage

- Operating a network is expensive
  - more than half the cost of a network
  - yet, operator error causes most outages

- Buggy software in the equipment
  - routers with 20+ million lines of code
  - cascading failures, vulnerabilities, etc.

- The network is “in the way”
  - especially a problem in data centers
  - ... and home networks
Traditional Computer Networks

Data plane:
- packet streaming
- forward, filter, buffer, mark, rate-limit, and measure packets
Traditional Computer Networks

Control plane:
distributed algorithms

track topology changes, compute routes, install forwarding rules
Traditional Computer Networks

Management plane: human time scale

collect measurements and configure the equipment
New Paradigm: Software Defined Networking (SDN)

logically-centralized control

smart, slow

API to the data plane (e.g., OpenFlow)

dumb, fast

switches
A Helpful Analogy
Mainframes

vertically integrated
closed, proprietary
slow innovation
small industry

Specialized Applications
Specialized Operating System
Specialized Hardware

horizontal open interfaces
rapid innovation
huge industry

Open Interface
Windows (OS) or Linux or Mac OS

Microprocessor
vertically integrated, closed, proprietary, slow innovation

horizontal, open interfaces, rapid innovation
How SDN works

The OpenFlow protocol
OpenFlow Switch specification

OpenFlow Switch

Secure Channel

Flow Table

OpenFlow Protocol

PC

Controller
Controller: Programmability

Controller Application

Network OS

- events from switches: topology changes, traffic statistics, arriving packets
- commands to switches: (un)install rules, query statistics, send packets
Reactive vs. Proactive

- **Reactive SDN:**
  switches send (first) packets to controller, then controller programs switch's flow table to handle rest of the flow
  - Problem: source of DoS on controller (packet-in event)

- **Proactive SDN:**
  Controller programs the switches proactively, according to its own knowledge of the network
  - Requires smarter approaches than just reacting to network events (global knowledge, discovery, updates...)

Flow Table Entry at Switch
“Type 0” OpenFlow Switch

Rule | Action | Stats
--- | --- | ---

Packet + byte counters

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

Switch Port | MAC src | MAC dst | Eth type | VLAN ID | IP Src | IP Dst | IP Prot | TCP sport | TCP dport
--- | --- | --- | --- | --- | --- | --- | --- | --- | ---
+ mask
Data-Plane: Simple Packet Handling

• Simple packet-handling rules
  – Pattern: match packet header bits
  – Actions: drop, forward, modify, send to controller
  – Priority: disambiguate overlapping patterns
  – Counters: #bytes and #packets

1. src=1.2.*.*, dest=3.4.5.* → drop
2. src = *.*.*.*, dest=3.4.*.* → forward(2)
3. src=10.1.2.3, dest=.*.*.* → send to controller
• Definition in progress
• Additional actions
  ➢ rewrite headers
  ➢ map to queue/class
  ➢ encrypt
• More flexible header
  ➢ allow arbitrary matching of first few bytes
• Support multiple controllers
  ➢ load-balancing and reliability
Example OpenFlow Applications

- Dynamic access control
- Seamless mobility/migration
- Server load balancing
- Network virtualization
- Using multiple wireless access points
- Energy-efficient networking
- Adaptive traffic monitoring
- Denial-of-Service attack detection

See http://www.openflow.org/videos/
E.g.: Dynamic Access Control

- Inspect first packet of a connection
- Consult the access control policy
- Install rules to block or route traffic
E.g.: Seamless Mobility/Migration

- See host send traffic at new location
- Modify rules to reroute the traffic
E.g.: Server Load Balancing

- Pre-install load-balancing policy
- Split traffic based on source IP

```
src=0*
src=1*
```
In-depth Example: Simple Repeater

- Simple Network Repeater
  - forward packets received on port 1 out 2 and vice versa
Controller (POX) (Pseudo)-Program

```python
def handle_packetIn(packet):
    out_port = 2
    if packet.in_port == 2:
        out_port = 1
    flow_mod = ofp_flow_mod()
    flow_mod.match = ofp_match()
    flow_mod.match.in_port = packet.in_port
    action = ofp_action_output()
    action.out_port = out_port
    flow_mod.action = [ action ]
    flow_mod.buffer_id = packet.buffer_id
    send(flow_mod)
```

Flow Table

<table>
<thead>
<tr>
<th>Priority</th>
<th>Pattern</th>
<th>Action</th>
<th>Counters</th>
</tr>
</thead>
<tbody>
<tr>
<td>DEFAULT</td>
<td>IN_PORT:1</td>
<td>OUTPUT:2</td>
<td>(0, 0)</td>
</tr>
<tr>
<td>DEFAULT</td>
<td>IN_PORT:2</td>
<td>OUTPUT:1</td>
<td>(0, 0)</td>
</tr>
</tbody>
</table>
Open Flow in the Wild

- **Open Networking Foundation**
  - Google, Facebook, Microsoft, Yahoo, Verizon, Deutsche Telekom, and many other companies

- **Commercial OpenFlow switches**
  - HP, NEC, Quanta, Dell, IBM, Juniper, ...

- **Network operating systems**
  - NOX, Beacon, Floodlight, POX, RYU

- **Network deployments**
  - Campuses, research backbone networks
  - Commercial deployments (e.g., Google backbone)
But... Heterogeneous Switches

- Number of packet-handling rules (TCAM/memory limits)
- Different OpenFlow version support
- Range of matches and actions (not all matches and actions are mandatory in the protocol)
- Multi-stage pipeline of packet processing (allowed but not defined in the standard)
- Vendor-specific features
- Offload some control-plane functionality (?)
• Controller is much slower than the switch
• Processing packets leads to delay and overhead
• Need to keep most packets in the “fast path”
Distributed Controller

For scalability and reliability

Partition and replicate state

... and: where to put the controller(s)?

Taking into account latency, resiliency, load balancing...
Testing and Debugging

• OpenFlow makes programming possible
  – Network-wide view at controller
  – Direct control over data plane

• Plenty of room for bugs
  – Still a complex, distributed system

• Need for testing techniques
  – Controller applications
  – Controller and switches
  – Rules installed in the switches
• **Controller APIs are low-level**
  – Thin veneer on the underlying hardware

• **Need better languages**
  – Composition of modules
  – Managing concurrency
  – Querying network state
  – Network-wide abstractions

• **Example:**
MiniNet
MiniNet

• Creates scalable SDN (up to hundreds of nodes) using OpenFlow, on a single PC

• Allows to quickly create, interact with and customize a SDN prototype with complex topologies, and can be used to emulate real networks - all on your PC

• Can work with any kind of OpenFlow controller

• Takes seconds to install

• Easy to program

• Of course, is an open source project
• Not only for teaching purposes!

• Used for the development and testing of networks
Conclusion

• SDN is revolutionizing networking

• Rethinking networking
  – open interfaces to the data plane
  – separation of control and data
  – leveraging techniques from distributed systems

• Significant momentum, many challenges
  – in both research and industry
Thank You