S7: Software Forwarding

Instructor:
Behnam Momeni

Fall 2018
Outline

- Click
- OpenFlow
Software-Based Routers

• Enabling innovation in networking research
• Software data planes

• Readings:
  • OpenFlow: Enabling Innovation in Campus Networks
  • The Click Modular Router

• Optional reading
  • RouteBricks: Exploiting Parallelism To Scale Software Routers
Click overview

- Modular architecture
  - Router = composition of modules
  - Router = data flow graph

- An element is the basic unit of processing

- Three key components of each element:
  - Ports
  - Configuration
  - Method interfaces
Simple Tee Element

**Diagram:**
- **Element Class:**
  - **Input Port:** Tee(2)
  - **Output Ports:**
  - **Configuration String:**
Two types of “connections”

- **Push**
  - Source element has finished processing
  - Sends it downstream
  - E.g., FromDevice

- **Pull**
  - Destination is ready to process
  - Initiates packet transfer
  - E.g., ToDevice
"Flow" of processing

FromDevice → Null
- receive packet p
- push(p)
- return

Null
- push(p)
- return
- enqueue p
- dequeue p and return it

Null → ToDevice
- pull()
- return p
- pull()
- return p
- ready to transmit
- send p
// Declare three elements...
src :: FromDevice(eth0);
ctr :: Counter;
sink :: Discard;
// ...and connect them together
src -> ctr;
ctr -> sink;

// Alternate definition using syntactic sugar
FromDevice(eth0) -> Counter -> Discard;
Other elements

- Packet Classification
- Scheduling
- Queueing
- Routing
- What you write…
Takeaways

• Click is a flexible modular router

• Shows that s/w x86 can get pretty good performance

• Extensible/modular

• Widely used in academia/research
  • Play with it!
Outline

• IP router design

• IP route lookup

• Click

• OpenFlow
Innovations in campus wiring closets

• Experiments we’d like to do
  • Mobility management
  • Network-wide energy management
  • New naming/addressing schemes
  • Network access control

• Problem with our network
  • Paths are fixed (by the network)
  • IP-only
  • Addresses dictated by DNS, DHCP, etc
  • No means to add our own processing
OpenFlow Switching

1. A way to run experiments in the networks we use everyday.
2. Bring GENI to college campuses.

A “pragmatic” compromise

*Allow researchers to run experiments in their network…  
…without requiring vendors to expose internal workings.*

Basics

*An Ethernet switch (e.g. 128-ports of 1GE)*  
*An open protocol to remotely add/remove flow entries*
Experimenter's Dream
(Vendor's Nightmare)

- Standard Network Processing
- User-defined Processing

Experimenter writes experimental code on switch/router
No obvious way

- Commercial vendor won’t open software and hardware development environment
  - Complexity of support
  - Market protection and barrier to entry

- Hard to build my own
  - Prototypes are flakey
  - Software only: Too slow
  - Hardware/software: Fanout too small (need >100 ports for wiring closet)
Furthermore, we want…

- Isolation: Regular production traffic untouched
- Virtualized and programmable: Different flows processed in different ways
- Equipment we can trust in our wiring closet
- Open development environment for all researchers (e.g. Linux, Verilog, etc).
- Flexible definitions of a flow
  - Individual application traffic
  - Aggregated flows
  - Alternatives to IP running side-by-side
  - …
OpenFlow Switching

OpenFlow Switch specification

OpenFlow Switch

Secure Channel

Flow Table

Controller

PC

OpenFlow Protocol
SSL

sw

hw
**Flow Table Entry**

“Type 0” OpenFlow Switch

<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

<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>

+ mask
What is nice

• Fits well with the TCAM abstraction

• Most vendors already have this

• They can just expose this without exposing internals
OpenFlow Usage Models

1. Experiments at the flow level
   - User-defined routing protocols
   - Admission control
   - Network access control
   - Network management
   - Energy management
   - VOIP mobility and handoff
   - ...
   
   • Experiment-specific controllers
   • Static or dynamic flow-entries

2. Experiments at the packet level
   - Slow: Controller handles packet processing
   - Fast: Redirect flows through programmable hardware
   - Modified routers, firewalls, NAT, congestion control…

3. Alternatives to IP
Example Apps

- Ethane
- Amy’s own OSPF
- VLAN
- VoIP for Mobile
- Support for non-IP
Example Experiment at the flow level

Mobility

Lots of interesting questions

- Management of flows
- Control of switches
- Access control of users and devices
- Tracking user location and motion
Experiments at the packet level

OpenFlow-enabled Commercial Switch

- Normal Software
- Normal Datapath
- Secure Channel
- Flow Table

Controller

PC

Laboratory

NetFPGA
OpenFlow Usage Models

1. Experiments at the flow level
2. Experiments at the packet level

3. Alternatives to IP
   - Flow-table is Layer-2 based
   - e.g. new naming and addressing schemes
   - ...

Network Management

Traffic Engineering
Performance
Security
Compliance
Resilience
Problem: Toolbox is bad!

Traffic Engineering
Performance
Security
Compliance
Resilience
Why: Toolbox is implicit in routers!

Motivation: Management is complex, expensive, fragile
Need: Direct control, expressive policy, network-wide views
Solution

- Separate out the “data” and the “control”
- Open interface between control/data planes
- Logically centralized views
  - Simplifies optimization/policy management
  - Network-wide visibility
Last Lecture: ONIX

Controller

E.g., ONIX, NOX, ...
Today: OpenFlow
Summary

• Building software routers by starting with:
  • closed, commercial routers vs.
  • commodity PCs
• Pros and cons?
Next Lecture

• Programming the Network/Evolution

• Readings:
  • P4: Read in full
  • Active Networks: skim