CS434/534: Topics in Networked (Networking) Systems

Basic Network Workflows; OpenFlow as a Datapath Programming Standard

Yang (Richard) Yang
Computer Science Department
Yale University
208A AK. Watson
Email: yry@cs.yale.edu

http://zoo.cs.yale.edu/classes/cs434/
Outline

- Admin and recap
- Journeys of a packet
- OpenFlow
- SDN programming
Start from next class (Wednesday next week), please start to read papers
How to Read

You May Think You Already Know How To Read, But...
You Spend a Lot of Time Reading

- Reading for classes
- Reviewing conference submissions
- Giving colleagues feedback
- Keeping up with your field
- Staying broadly educated
- Transitioning into a new areas
- Learning how to write better papers 😊
Keshav’s Three-Pass Approach: Step 1

- A ten-minute scan to get the general idea
  - Title, abstract, and introduction
  - Section and subsection titles
  - Conclusion and bibliography

- What to learn: the five C’s
  - Category: What type of paper is it?
  - Context: What body of work does it relate to?
  - Correctness: Do the assumptions seem valid?
  - Contributions: What are the main research contributions?
  - Clarity: Is the paper well-written?

- Decide whether to read further...
Keshav’s Three-Pass Approach: Step 2

- **A more careful, one-hour reading**
  - Read with greater care, but ignore details like proofs
  - Figures, diagrams, and illustrations
  - Mark relevant references for later reading

- **Grasp the content of the paper**
  - Be able to summarize the main idea
  - Identify whether you can (or should) fully understand

- **Decide whether to**
  - Abandon reading in greater depth
  - Read background material before proceeding further
  - Persevere and continue for a third pass
Keshav’s Three-Pass Approach: Step 3

- Several-hour virtual re-implementation of the work
  - Making the same assumptions, recreate the work
  - Identify the paper’s innovations and its failings
  - Identify and challenge every assumption
  - Think how you would present the ideas yourself
  - Jot down ideas for future work

- When should you read this carefully?
  - Reviewing for a conference or journal
  - Giving colleagues feedback on a paper
  - Understanding a paper closely related to your research
  - Deeply understanding a classic paper in the field
Other Tips for Reading Papers

- Read at the right level for what you need
  - “Work smarter, not harder”
- Read at the right time of day
  - When you are fresh, not sleepy
- Read in the right place
  - Where you are not distracted, and have enough time
- Read actively
  - With a purpose (what is your goal?)
  - With a pen or computer to take notes
- Read critically
  - Think, question, challenge, critique, ...
Recap

- Network systems involve many protocols

- Network systems involve many boxes

- One key goal of this class: unifying the protocols and boxes
Recap: Topics

- Data centers (cloud) and challenges
- IoT and challenges
- Data analytics and challenges
Recap: Main Design Decisions/Principles

- Packets (divide)
- Layered architecture
  - End-to-end argument
  - Logical communications and physical communications
Outline

- Admin and recap
- Journeys of a packet
- OpenFlow
- SDN programming
Exercise: Fetch Class Homepage
Name -> IP Address

- **Why IP address**
  - Uniform, compact format (32 bits)
  - IP address gives a locator (another locator in daily life?)

- **IP address assigned to each interface**
  - Try `ifconfig`
Transport (L4): Connection, Message, Packets

- Transport breaks message into packets (frames)

- Host uses ports to distinguish different connections (conversations)
  - Typically src IP + dst IP + src Port + dst Port defines connection
  - Try netstat -a -p tcp

- A connection has setup and tear down phases
Network (L3) and Link (L2) Layers

- Network layer finds a path for each packet
- Link layer traverses each link of the path
Outline

- Admin and recap
- Journeys of a packet
  - Top down
  - Bottom up and use cases
Link (L2) Layer: Setting

- Link layer service provided by (L2) switches
- Interface
  send: A -> B

- Q: How does the switch implement the service?

Source: A
Dest: B
Modern L2 and Switch Table

- Modern L2 uses switch table to realize its service

when frame received at L2 switch:
1. index its switch table using L2 destination address
2. if (entry found for destination) {
   if (destination on interface from which frame arrived)
     drop frame
   else
     forward frame on interface indicated by entry
}

Q: How to handle unknown host?

<table>
<thead>
<tr>
<th>MAC addr</th>
<th>interface</th>
<th>TTL</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>1</td>
<td>60</td>
</tr>
</tbody>
</table>
when frame received at link-layer switch:
1. **record** incoming link, MAC address of sending host
2. index switch table using MAC destination address
3. if (entry found for destination) {
   if (destination on interface from which frame arrived)
      drop frame
   else
      forward frame on interface indicated by entry
} 
else
   **flood** /* forward on all interfaces except arriving interface */
Example: Flooding, Learning

- frame destination, B, location unknown: flood

- destination A location known: selectively send on just one link

<table>
<thead>
<tr>
<th>MAC addr</th>
<th>interface</th>
<th>TTL</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>1</td>
<td>60</td>
</tr>
<tr>
<td>B</td>
<td>4</td>
<td>60</td>
</tr>
</tbody>
</table>

switch table (initially empty)
A layer-2 domain may need multiple switches.

Exercise: Assume A sends to G. Will the flooding+learning alg work to build switch tables for the switches in the above network?

For what kind of switch topology does the alg work?
Extension: LAN -> VLAN

Setting:
- Mixed office environment
- Per switch per group
  - expensive
- Shared switch
  - single broadcast domain
  - security/privacy, efficiency issues
Extension: VLANs

port-based VLAN: switch ports grouped (by switch management software) so that single physical switch …..

Discussion: what does switch table look like? How to configure?

… operates as multiple virtual switches
Problem:
- a VLAN may span multiple switches,
- frames forwarded within VLAN between switches can’t be vanilla 802.1 frames (must carry VLAN ID info)
- 802.1q protocol adds/removed additional header fields for frames forwarded between switches
802.1Q VLAN Frame Format

Discussion: what does switch table look like?
Summary: L2 Domain

- Multiple switches form a large L2 domain
- There are methods to divide the large domain into virtual L2 domains
- Switch tables at switches connect the devices

Discussion:
  - Can we use the approach to cover the whole Internet or at least a large network?
Outline

- Admin and recap
- Journeys of a packet
  - Top down
  - Bottom up and use cases
    - L2: Basic L2, L2 with VLAN
    - L3
L3 Forwarding

Main idea

- Use routing to replace flooding to forward packets to the right direction
  - Routing table computed by routing protocols

- Question: why we cannot use L2 (MAC) address for L3 forwarding?
Comparison of L2 (MAC) and L3 (IP) Addresses

- L2 (MAC) address can be an identifier
  - dedicated to a device
  - flat

- L3 (IP) address is locator
  - address depends on network to which an interface is attached
  - must support aggregation for scalability (does phone# support aggregation?)

- L3 (MAC) address does not need to be globally unique, but the current assignment ensures uniqueness

- L3 (IP) address needs to be globally unique
Classless InterDomain Routing (CIDR) Address: Aggregation

- A CIDR address partitions an IP address into two parts
  - A prefix representing the network portion, and the rest (host part)
  - address format: \texttt{a.b.c.d/x}, where \( x \) is \# bits in network portion of address

\begin{center}
\begin{tikzpicture}
  \node (network) at (0,0) {network part};
  \node (host) at (3,0) {host part};
  \node[draw,circle,inner sep=2pt] (networkpart) at (0,0) {11001000};
  \node[draw,circle,inner sep=2pt] (hostpart) at (3,0) {00000000};
  \draw[->,blue] (networkpart) -- (network);
  \draw[<-] (network) -- (networkpart);
  \draw[->,blue] (hostpart) -- (host);
  \draw[<-] (host) -- (hostpart);
\end{tikzpicture}
\end{center}

200.23.16.0/23

Some systems use mask (1’s to indicate network bits), instead of the /x format
Summary: L3 Forwarding

Main ideas

- Use routing to replace flooding to forward packets to the right direction

- Use IP address aggregation to reduce table size
Exercise

- Show the routing table of a zoo host
- Query the routing table of a BGP router
  - https://us.ntt.net/support/looking-glass/
L3: Example 1 (same network): A -> B

- Look up dest address in routing table
  - find dest is on same net (what is the matching alg?)

- Hand datagram to L2 to send inside a link-layer frame
ARP Protocol

- Issue: Need to convert L3 address to L2 address

- Current solution:
  - Data structure: each host has an ARP table mapping IP to L2
    - `arp -a
      IP -> MAC`
  - Protocol: a broadcast protocol
    - Client broadcasts query frame, containing queried IP address
    - Node with queried IP receives ARP frame, replies its MAC address
ARP Format and Features

- **Query**: Layer 2 (Link layer) broadcast: destination \texttt{ff:ff:ff:ff:ff:ff} to be received by all hosts at the same local network.
- **Response**: Host with the MAC returns its MAC if it has the query IP.
- **Gratuitous ARP**: A host sends this message to update other devices if it changes MAC.
A -> B in Layer 2

After A finds the link-layer (MAC) address of B, it constructs the L2 frame:

<table>
<thead>
<tr>
<th>frame source, dest address</th>
<th>datagram source, dest address</th>
</tr>
</thead>
<tbody>
<tr>
<td>A’s MAC addr</td>
<td>B’s MAC addr</td>
</tr>
</tbody>
</table>

```
223.1.1.1
223.1.1.2
223.1.1.3
223.1.1.4
223.1.2.1
223.1.2.2
223.1.2.9
223.1.3.1
223.1.3.2
223.1.3.27
```

---

A

B

---

XR

---

223.1.1.1

223.1.1.2

223.1.1.3

223.1.1.4

223.1.2.1

223.1.2.2

223.1.2.9

223.1.3.1

223.1.3.2

223.1.3.27
Datapath: Example 2 (Different Networks): A→E

- Look up dst address in routing table
  - routing table: next hop router to dest is 223.1.1.4

- Hand datagram to link layer to send to router 223.1.1.4 inside a link-layer frame
Datapath: Example 2 (Different Networks): A→E

- **look up dest address in router's forwarding table**
  - E on same network as router's interface 223.1.2.9

- **link layer sends datagram to 223.1.2.3 inside link-layer frame via interface 223.1.2.9**
Outline

- Admin and recap
- Journeys of a packet
  - Top down
  - Bottom up and use cases
    - L2: Basic L2, L2 with VLAN
    - L3: L3 lookup, L3/L2 address resolution
    - L4: NAT, NLB
Setting: Network Address Translation/Load Balancing

Setting:
- a single public/front end IP, called Virtual IP
- multiple servers w/ their own IPs
Table 1: an example of virtual server rules

<table>
<thead>
<tr>
<th>Protocol</th>
<th>Virtual IP Address</th>
<th>Port</th>
<th>Real IP Address</th>
<th>Port</th>
<th>Weight</th>
</tr>
</thead>
<tbody>
<tr>
<td>TCP</td>
<td>202.103.106.5</td>
<td>80</td>
<td>172.16.0.2</td>
<td>80</td>
<td>1</td>
</tr>
<tr>
<td>TCP</td>
<td>202.103.106.5</td>
<td>21</td>
<td>172.16.0.3</td>
<td>21</td>
<td>1</td>
</tr>
</tbody>
</table>

NLB/NAT

Client

request packet

source ip 206.183.42.40
destination ip 192.168.0.100

VIP: 192.168.0.100
RIP: 192.168.0.201

active load balancer

primary heartbeat

backup heartbeat

backup load balancer

Server 1

RIP: 192.168.0.205

Server n

RIP: 192.168.0.206
NLB/NAT Flow
## Summary

<table>
<thead>
<tr>
<th>Function</th>
<th>Data Structure</th>
<th>How to use</th>
<th>How currently populated?</th>
</tr>
</thead>
<tbody>
<tr>
<td>L2 basic</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>L2 VLAN</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>L3 lookup</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ARP</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>NAT/LB</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Table, Table, Table

- Various of tables
- Not only tables, only match-action tables
Outline

- Admin and recap
- Journeys of a packet
- SDN basic idea and OpenFlow
Distributed Computing => Logically Centralized Control

- Distributed computing is hard, e.g.,
  - FLP Impossibility Theorem
  - Arrow’s Impossibility Theorem

- Need central configuration anyway
SDN: Separation of data and control planes

Traditional

Control
Datapath

Control
Datapath

SDN

Control

standard control protocol

Datapath

Datapath

Datapath
What is OpenFlow?

- The first standard
  - data path abstraction of devices
  - communications protocol defined between controller and device.
OpenFlow Datapath Abstraction

- Device contain flow tables

- Each flow table is a set of flow entries

- A flow entry consists of
  - matching condition
  - action
## OpenFlow: Flow entry/rule

### Match Fields
- **match fields**: to match against packets. These consist of the ingress port and packet headers, and optionally other pipeline fields such as metadata specified by a previous table.
- **priority**: matching precedence of the flow entry.
- **counters**: updated when packets are matched.
- **instructions**: to modify the action set or pipeline processing.
- **timeouts**: maximum amount of time or idle time before flow is expired by the switch.
- **cookie**: opaque data value chosen by the controller. May be used by the controller to filter flow entries affected by flow statistics, flow modification and flow deletion requests. Not used when processing packets.
- **flags**: flags alter the way flow entries are managed, for example the flag `OFPFF_SEND_FLOW_REM` triggers flow removed messages for that flow entry.
OpenFlow: Match Fields

Match Fields | Action | Stats

Switch Port  | VLAN ID | VLAN pcp | MAC src | MAC dst | Eth type | IP Src | IP Dst | IP ToS | IP Prot | L4 sport | L4 dport

+ mask what fields to match

Source: Scott Shenker, UC Berkeley
## Examples

### L2 basic

<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>

### L3 load balancer

<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>*</td>
<td></td>
</tr>
</tbody>
</table>

### Firewall (e.g., block SSH)

<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>

Source: Scott Shenker, UC Berkeley
## 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 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>*</td>
<td>00:1f..</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>port6, port7, port9</td>
</tr>
</tbody>
</table>
OpenFlow: Action

1. Forward packet to zero or more ports
2. Encapsulate and forward to controller
3. Send to normal processing pipeline
4. Modify Fields
5. Any extensions you add!

+ mask what fields to match

Source: Scott Shenker, UC Berkeley
Offline Exercise: OF using Mininet

- An environment to play with OF
  - Download: http://mininet.org/download/
  - Tutorial: https://github.com/mininet/openflow-tutorial/wiki
No match is found???

- A table-miss flow entry to process table misses
- May send packets to the controller, drop packets or direct packets to a subsequent table.
Example OF Test Commands

- `sudo mn --topo single,3 --mac --switch ovsk --controller remote`
- `sh ovs-ofctl dump-flows s1`
- `sh ovs-ofctl add-flow s1 in_port=1,actions=output:2`
- `sh ovs-ofctl add-flow s1 in_port=2,actions=output:1`
- `sh ovs-ofctl del-flows s1`
- `sh ovs-ofctl add-flow s1 "priority=0,action=normal"`
- `sh ovs-ofctl add-flow s1 "priority=100,eth_type=0x800,ip_dst=10.0.0.1,action=drop"`
- `sh ovs-ofctl add-flow s1 "priority=100,eth_type=0x806,dl_dst=00:00:00:00:00:02,action=drop"`
For next time...