The Network Layer: Data Plane

Raj Jain
Washington University in Saint Louis
Saint Louis, MO 63130
Jain@wustl.edu
Audio/Video recordings of this lecture are available on-line at:
http://www.cse.wustl.edu/~jain/cse473-21/
Overview

1. Network Layer Basics
2. What’s inside a router?
3. Forwarding Protocols: IPv4, DHCP, NAT, IPv6
4. Software Defined Networking

Note: This class lecture is based on Chapter 4 of the textbook (Kurose and Ross) and the figures provided by the authors.
Network Layer Basics

1. Forwarding and Routing
2. Connection Oriented Networks: ATM Networks
3. Classes of Service
4. Router Components
5. Packet Queuing and Dropping
Forwarding and Routing

- **Forwarding**: Input link to output link via Address prefix lookup in a table.
- **Routing**: Making the Address lookup table
- **Longest Prefix Match**

<table>
<thead>
<tr>
<th>Prefix</th>
<th>Next Router</th>
<th>Interface</th>
</tr>
</thead>
<tbody>
<tr>
<td>126.23.45.67/32</td>
<td>125.200.1.1</td>
<td>1</td>
</tr>
<tr>
<td>128.272.15/24</td>
<td>125.200.1.2</td>
<td>2</td>
</tr>
<tr>
<td>128.272/16</td>
<td>125.200.1.1</td>
<td>1</td>
</tr>
</tbody>
</table>

Ref: Optional Homework: R3 in the textbook

Washington University in St. Louis  
http://www.cse.wustl.edu/~jain/cse473-21/  
©2021 Raj Jain
Network Service Models

- Guaranteed Delivery: No packets lost
- Bounded delay: Maximum delay
- In-Order packet delivery: Some packets may be missing
- Guaranteed minimal throughput
- Guaranteed maximum jitter: Delay variation
- Security Services (optional in most networks)
- ATM offered most of these
- IP offers none of these \(\Rightarrow\) Best effort service (Security is optional)

Optional Homework: R4, R5 in the textbook
What's Inside a Router?

- **Input Ports**: receive packets, lookup address, queue
  - Use **Content Addressable Memories** (CAMs) and caching
- **Switch Fabric**: Send from input port to output port
- **Output Ports**: Queuing, transmit packets
Types of Switching Fabrics

- Memory
- Bus
- Crossbar

Student Questions
Where Does Queuing Occur?

- If switching fabric is slow, packets wait on the input port.
- If switching fabric is fast, packets wait for output port
  ⇒ Queueing (Scheduling) and drop policies
- Queueing: First Come First Served (FCFS), Weighted Fair Queueing
**Ideal Buffering**

- Flow Control Buffering = RTT*Transmission Rate
- Buffer = RTT*Transmission Rate/$\sqrt{(# \text{ of TCP flows})}$
Packet Dropping Policies

- **Drop-Tail**: Drop the arriving packet
- **Random Early Drop (RED)**: Drop arriving packets even before the queue is full
  - Routers measure average queue and drop incoming packet with certain probability
  - ⇒ **Active Queue Management (AQM)**

![Diagram showing probability of drop vs average queue](image-url)
Packet at the head of the queue is waiting
⇒ Other packets can not be forwarded even if they are going to other destination
Network Layer Basics: Review

1. Forwarding uses routing table to find output port for datagrams using **longest prefix match**. Routing protocols make the table.
2. IP provides only **best effort** service (KISS).
4. Datagrams may be dropped even if the queues are not full (**Random early drop**).
5. Queueing at input may result in **head of line blocking**.

Ref: Read Sections 4.1, 4.2, full, Page 305-329 of the textbook. Try R1 through R16
Washington University in St. Louis  http://www.cse.wustl.edu/~jain/cse473-21/  ©2021 Raj Jain
Forwarding Protocols

1. IPv4 Datagram Format
2. IP Fragmentation and Reassembly
3. IP Addressing
4. Network Address Translation (NAT)
5. Universal Plug and Play
6. Dynamic Host Control Protocol (DHCP)
7. IPv6
**IP Datagram Format**

<table>
<thead>
<tr>
<th>Field</th>
<th>Length</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Version</td>
<td>4b</td>
<td>in 4B words</td>
</tr>
<tr>
<td>Header Length</td>
<td>4b</td>
<td>in 4B words</td>
</tr>
<tr>
<td>Type of Service</td>
<td>8b</td>
<td></td>
</tr>
<tr>
<td>Total Length</td>
<td>16b</td>
<td>in Bytes</td>
</tr>
<tr>
<td>Datagram ID</td>
<td>16b</td>
<td></td>
</tr>
<tr>
<td>Rsvd</td>
<td>1b</td>
<td>Don’t Fragment</td>
</tr>
<tr>
<td>More Fragments</td>
<td>1b</td>
<td></td>
</tr>
<tr>
<td>Fragment Offset</td>
<td>13b</td>
<td>in 8B blocks</td>
</tr>
<tr>
<td>Upper Layer</td>
<td>8b</td>
<td>Protocol</td>
</tr>
<tr>
<td>Flags</td>
<td>8b</td>
<td></td>
</tr>
<tr>
<td>Header Checksum</td>
<td>16b</td>
<td></td>
</tr>
<tr>
<td>Source IP Address</td>
<td>32b</td>
<td></td>
</tr>
<tr>
<td>Destination IP Address</td>
<td>32b</td>
<td></td>
</tr>
<tr>
<td>Options</td>
<td>Variable</td>
<td></td>
</tr>
<tr>
<td>Padding</td>
<td>Variable</td>
<td></td>
</tr>
<tr>
<td>Payload</td>
<td>Variable</td>
<td></td>
</tr>
</tbody>
</table>

Flags:
- 1=ICMP
- 6=TCP
- 17=UDP

Multiple of 4B
IP Fragmentation Fields

- Header length: in units of 32-bit words
- Data Unit Identifier (ID)
  - Sending host puts an identification number in each datagram
- Total length: Length of user data plus header in bytes
- Fragment Offset - Position of fragment in original datagram
  - In multiples of 8 byte blocks
- More fragments flag
  - Indicates that this is not the last fragment
- Datagrams can be fragmented/refragmented at any router
- Datagrams are reassembled only at the destination host
IP Fragmentation and Reassembly

Example

- 4000 byte datagram
- Maximum Transmission Unit (MTU) = 1500 bytes

Fragment data ≥ 8 Bytes
IP Header ≤ 60 Bytes
MTU ≥ 68 Bytes

One large datagram becomes several smaller datagrams

1480 bytes in data field

offset = 1480/8

©2021 Raj Jain

http://www.cse.wustl.edu/~jain/cse473-21/
[8 points] Consider sending a 2400-byte datagram into a link that has an MTU of 720 bytes. Suppose the original datagram is stamped with the identification number 422. How many fragments are generated? What are the values in the various fields in the IP datagram(s) generated related to fragmentation?
**IP Address Classes**

- **Class A:**
  - Local: 0110 24 bits
  - Network: 1 Network

- **Class B:**
  - Local: 10 Network 16 bits
  - Network: 2

- **Class C:**
  - Local: 110 Network 8 bits
  - Network: 3

- **Class D:**
  - Host Group (Multicast): 1110 28 bits

- **Class E:**
  - Future use: 11110

- Local = Subnet + Host (Variable length)
- All IP hosts have a 32-bit address. 128.10.0.1
  = 1000 0000 0000 1010 0000 0000 0000 0001
- All hosts on a network have the same network prefix
All hosts on a subnetwork have the same prefix. Position of the prefix is indicated by a “subnet mask”

Example: First 23 bits = subnet
Address: 10010100 10101000 00010000 11110001
Mask: 11111111 11111111 11111110 00000000
\[ \text{.AND.} \quad 10010100 \ 10101000 \ 00010000 \ 00000000 \]

Network

Subnet 1

\[ \cdots \]

Subnet 2

Subnet n
IP addressing: CIDR

- CIDR: Classless InterDomain Routing
  - Subnet portion of address of arbitrary length
  - Address format: a.b.c.d/x, where x is # bits in subnet portion of address
  - All 1’s in the host part is used for subnet broadcast
  - All 0’s in the host part was meant as “subnet address” but not really used for anything. Some implementation allow it to be used as host address. Some don’t. Better to avoid it.

```
11001000  00010111 00010000  00000000
```

```
subnet part
11001000 00010111 00010000 00000000
```

```
host part
```

200.23.16.0/23
[22 points] Consider a router that interconnects 3 subnets: Subnet 1, Subnet 2, and Subnet 3. Suppose all of the interfaces in each of these three subnets are required to have the prefix 223.1.17/24. Also suppose that Subnet 1 is required to support up to 61 interfaces, Subnet 2 is to support up to 96 interfaces, and Subnet 3 is to support up to 16 interfaces. Provide three network address prefixes (of the form a.b.c.d/x) that satisfy these constraints. Use adjacent allocations. For each subnet, also list the subnet mask to be used in the hosts.
Forwarding an IP Datagram

- Delivers datagrams to destination network (subnet)
- Routers maintain a “routing table” of “next hops”
- Next Hop field does not appear in the datagram

Table at R2:

<table>
<thead>
<tr>
<th>Destination</th>
<th>Next Hop</th>
</tr>
</thead>
<tbody>
<tr>
<td>Net 1</td>
<td>Forward to R1</td>
</tr>
<tr>
<td>Net 2</td>
<td>Deliver Direct</td>
</tr>
<tr>
<td>Net 3</td>
<td>Deliver Direct</td>
</tr>
<tr>
<td>Net 4</td>
<td>Forward to R3</td>
</tr>
</tbody>
</table>
Route Aggregation

- Can combine two or more prefixes into a shorter prefix
- ISPs-R-Us has a more specific route to organization 1

```
Organization 0
  200.23.16.0/23

Organization 2
  200.23.20.0/23

Organization 7
  200.23.30.0/23

Organization 1
  200.23.18.0/23

Fly-By-Night-ISP

ISPs-R-Us

“Send me anything with addresses beginning 200.23.16.0/20”

“Send me anything with addresses beginning 199.31.0.0/16 or 200.23.18.0/23”

Internet
```

Ref: Section 4.3.3. Try R27, R28
http://www.cse.wustl.edu/~jain/cse473-21/
### “Route Print” Command in Windows

**MAC:** netstat -rn

### Interface List
- 0x1 ...................... MS TCP Loopback interface
- 0x2 ...00 16 eb 05 af c0 ...... Intel(R) WiFi Link 5350 - Packet Scheduler Miniport
- 0x3 ...00 1f 16 15 7c 41 ...... Intel(R) 82567LM Gigabit Network Connection - Packet Scheduler Miniport
- 0x40005 ...00 05 9a 3c 78 00 ...... Cisco Systems VPN Adapter - Packet Scheduler Miniport

### Active Routes:

<table>
<thead>
<tr>
<th>Network Destination</th>
<th>Netmask</th>
<th>Gateway</th>
<th>Interface</th>
<th>Metric</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.0.0.0</td>
<td>0.0.0.0</td>
<td>192.168.0.1</td>
<td>192.168.0.108</td>
<td>10</td>
</tr>
<tr>
<td>0.0.0.0</td>
<td>0.0.0.0</td>
<td>192.168.0.1</td>
<td>192.168.0.106</td>
<td>10</td>
</tr>
<tr>
<td>127.0.0.0</td>
<td>255.0.0.0</td>
<td>127.0.0.1</td>
<td>127.0.0.1</td>
<td>1</td>
</tr>
<tr>
<td>169.254.0.0</td>
<td>255.255.0.0</td>
<td>192.168.0.106</td>
<td>192.168.0.106</td>
<td>20</td>
</tr>
<tr>
<td>192.168.0.0</td>
<td>255.255.255.0</td>
<td>192.168.0.106</td>
<td>192.168.0.106</td>
<td>10</td>
</tr>
<tr>
<td>192.168.0.106</td>
<td>255.255.255.255</td>
<td>127.0.0.1</td>
<td>127.0.0.1</td>
<td>10</td>
</tr>
<tr>
<td>192.168.0.108</td>
<td>255.255.255.255</td>
<td>127.0.0.1</td>
<td>127.0.0.1</td>
<td>10</td>
</tr>
<tr>
<td>192.168.0.255</td>
<td>255.255.255.255</td>
<td>192.168.0.106</td>
<td>192.168.0.106</td>
<td>10</td>
</tr>
<tr>
<td>224.0.0.0</td>
<td>240.0.0.0</td>
<td>192.168.0.106</td>
<td>192.168.0.106</td>
<td>10</td>
</tr>
<tr>
<td>224.0.0.0</td>
<td>240.0.0.0</td>
<td>192.168.0.106</td>
<td>192.168.0.106</td>
<td>10</td>
</tr>
<tr>
<td>Default Gateway:</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Persistent Routes:
- None

### Note:
- 127.0.0.1 = Local Host
- 224.x.y.z = Multicast on local LAN

---

**Student Questions**

- Adr & mask = Dest
  - Match
- Longest Prefix match
  - is used
- Metric: Lower is better

---

**Washington University in St. Louis**


©2021 Raj Jain
Lab 4A: Routing Table

- [8 Points] Use “Route Help” in Windows (or man route in MAC) to learn the route command
- Ping [www.google.com](http://www.google.com) to find its address
- Make sure that you have two active interfaces preferably connected to different routers. For example, create a 2nd interface by connecting a smart phone hot spot via USB. Or by connecting to a router in our lab during TA hours
- Print route table
- Trace route to [www.google.com](http://www.google.com) using tracert
- Modify the routing table so that the other interface will be used.
- Note the command you used to modify the routing table
- Print the new routing table
- Trace route to the same numeric address for [www.google.com](http://www.google.com) as before. Submit underlined items.
Lab 4A Hints

- A host with two interfaces going to the same router:

- Trace route result will not change even if you change the interface.

### IPv4 Route Table

<table>
<thead>
<tr>
<th>Network Destination</th>
<th>Netmask</th>
<th>Gateway</th>
<th>Interface</th>
<th>Metric</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.0.0.0</td>
<td>0.0.0.0</td>
<td>192.168.0.1</td>
<td>192.168.0.152</td>
<td>55</td>
</tr>
<tr>
<td>0.0.0.0</td>
<td>0.0.0.0</td>
<td>192.168.0.1</td>
<td>192.168.0.151</td>
<td>25</td>
</tr>
</tbody>
</table>
Lab 4A Hints (Cont)

- If you have two routers, you can see the effect in trace route. One way to get two routers is to use your cell phone hot spot:

- WiFi on phone should be disabled to ensure that it does not forward the traffic to the same home router.
Another way to get two routers is to use another router. We have placed an extra router in our lab.

---

IPv4 Route Table

<table>
<thead>
<tr>
<th>Active Routes:</th>
<th>Network Destination</th>
<th>Netmask</th>
<th>Gateway</th>
<th>Interface Metric</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>0.0.0.0</td>
<td>0.0.0.0</td>
<td>192.168.0.1</td>
<td>192.168.0.151</td>
</tr>
<tr>
<td></td>
<td>0.0.0.0</td>
<td>0.0.0.0</td>
<td>172.20.10.1</td>
<td>172.20.10.2</td>
</tr>
</tbody>
</table>

---

Student Questions
Lab 4A Hints (Cont)

- WWW.google.com may have different IP addresses on different networks and so trace route to the same numeric address.
- WUSTL VPN rejects all traffic not going to WUSTL. So it can not be used as the 2\textsuperscript{nd} interface.
- The new metric assigned by the route command may not be what you specified. So always check using route print.
Lab 4A Hints (Cont)

A. Use “route help” to learn the route command
- **Windows:** route help
- **Linux:** route help
- **MAC:**
  - man netstat
  - man route

B. Ping [www.google.com](http://www.google.com) to find its address
- ping [www.google.com](http://www.google.com)

C. Print the new routing table
- **Windows:**
  - route print
- **Linux:**
  - route
- **MAC:**
  - netstat -nr

D. Modify routing tables
- **Windows:**
  - route add/delete/change
- **Linux:**
  - route add/del
- **MAC:**
  - sudo route –nv add

E. Verify using tracert
- **Windows:**
  - tracert
- **Linux:**
  - traceroute
- **MAC:**
  - traceroute
Private Addresses

- Any organization can use these inside their network
  Can’t go on the internet. [RFC 1918]
- 10.0.0.0 - 10.255.255.255 (10/8 prefix)
- 172.16.0.0 - 172.31.255.255 (172.16/12 prefix)
- 192.168.0.0 - 192.168.255.255 (192.168/16 prefix)
Network Address Translation (NAT)

- Private IP addresses 192.168.x.x
- Can be used by anyone inside their networks
- Cannot be used on the public Internet
- NAT overwrites source addresses on all outgoing packets and overwrites destination addresses on all incoming packets
- Only outgoing connections are possible

192.168.0.2 192.168.0.3
192.168.0.4 192.168.0.5

128.252.73.216 Internet

Student Questions
Universal Plug and Play

- NAT needs to be manually programmed to forward external requests
- UPnP allows hosts to request port forwarding
- Both hosts and NAT should be UPnP aware
- Host requests forwarding all port xx messages to it
- NAT returns the public address and the port #.
- Host can then announce the address and port # outside
- Outside hosts can then reach the internal host (server)
[4 points] Consider a home network of 3 computers connected to the Internet via a NAT router. Suppose the ISP assigns the router the address 23.34.112.235 and that the network address of the home network is 192.168.1/29.

A. Assign addresses to all interfaces in the home network starting with the lowest possible address.

B. What is the subnet mask for the home computers?

C. Suppose each host has two ongoing TCP connections, all to port 80 at host 128.119.40.86. Provide the six corresponding entries in the NAT translation table. Both NAT and computers use source ports starting at 3000.
DHCP

- Dynamic Host Control Protocol
- Allows hosts to get an IP address automatically from a server
- Do not need to program each host manually
- Each allocation has a limited “lease” time
- Can reuse a limited number of addresses
- Hosts broadcast “Is there a DHCP Server Here?”
  Sent to 255.255.255.255
- DHCP servers respond
**DHCP Example**

DHCP server: 223.1.2.5

---

**DHCP discover**
- 
  src: 0.0.0.0, 68
  dest: 255.255.255.255, 67
  yiaddr: 0.0.0.0
  transaction ID: 654

---

**DHCP offer**
- 
  src: 223.1.2.5, 67
  dest: 255.255.255.255, 68
  yiaddr: 223.1.2.4
  transaction ID: 654
  Lifetime: 3600 secs

---

**DHCP request**
- 
  src: 0.0.0.0, 68
  dest: 255.255.255.255, 67
  yiaddr: 223.1.2.4
  transaction ID: 655
  Lifetime: 3600 secs

---

**DHCP ACK**
- 
  src: 223.1.2.5, 67
  dest: 255.255.255.255, 68
  yiaddr: 223.1.2.4
  transaction ID: 655
  Lifetime: 3600 secs
[15 points] Download the Wireshark traces from http://gaia.cs.umass.edu/wireshark-labs/wireshark-traces.zip

Open dhcp-ethereal-trace-1 in Wireshark. Select View → Expand All. Answer the following questions:

1. Examine Frame 2 marked DHCP.
   A. What transport protocol and destination port # is used by DHCP?
   B. What are the source and destination IP addresses for this frame and why?
   C. What is the Type-Length-Value for the DHCP Discover option?

2. Examine Frames 4, 5, 6 to find Type-Length-Value for:
   A. DHCP Offer
   B. DHCP Request
   C. DHCP Ack
Lab 4B: DHCP (Cont)

3. Examine Frame 4:
   A. What IP address was assigned by the DHCP server?
   B. What IP address is this frame addressed to and why?
   C. What other information was provided by the DHCP server?
      1. Subnet Mask:
      2. Default Gateway:
      3. DNS1:
      4. DNS2:
      5. Domain Name:
      6. Lease Time:

4. Examine Frame 5 and find what preferred IP address was requested by the client?
IPv6

- Shortage of IPv4 addresses ⇒ Need larger addresses
- IPv6 was designed with 128-bit addresses
  \[2^{128} = 3.4 \times 10^{38}\] addresses
  ⇒ \(665 \times 10^{21}\) addresses per sq. m of earth surface
- If assigned at the rate of \(10^6/\mu s\), it would take 20 years
- **Dot-Decimal**: 127.23.45.88
- **Colon-Hex**: FEDC:0000:0000:0000:3243:0000:0000:ABCD
  - Can skip leading zeros of each word
  - Can skip one sequence of zero words, e.g.,
    FEDC::3243:0000:0000:ABCD
    ::3243:0000:0000:ABCD
  - Can leave the last 32 bits in dot-decimal, e.g., ::127.23.45.88
  - Can specify a prefix by /length, e.g., 2345:BA23:0007::/50
**IPv6 Header**

- **IPv6:**
  - Version (4b)
  - Traffic Class (8b)
  - Flow Label (20b)
  - Payload Length (16b)
  - Next Header (8b)
  - Hop Limit (8b)
  - Source Address (128b)
  - Destination Address (128b)

- **IPv4:**
  - Version | IHL | Type of Service | Total Length
  - Identification | Flags | Fragment Offset
  - Time to Live | Protocol | Header Checksum
  - Source Address
  - Destination Address
  - Options
  - Padding

---

**Student Questions**
**IPv6 vs. IPv4**

- 1995 vs. 1975
- IPv6 only twice the size of IPv4 header
- Only version number has same position and meaning as in IPv4
- Removed: header length, type of service, identification, flags, fragment offset, header checksum ⇒ No fragmentation
- Datagram length replaced by payload length
- Protocol type replaced by next header
- Time to live replaced by hop limit
- Added: Priority and flow label
- All fixed size fields.
- No optional fields. Replaced by extension headers.
- 8-bit hop limit = 255 hops max (Limits looping)
- Next Header = 6 (TCP), 17 (UDP)
**IPv4 to IPv6 Transition**

- **Dual Stack:** Each IPv6 router also implements IPv4. IPv6 is used only if source host, destination host, and all routers on the path are IPv6 aware.

- **Tunneling:** The last IPv6 router puts the entire IPv6 datagram in a new IPv4 datagram addressed to the next IPv6 router = **Encapsulation**

---

**Physical view:**

- S
- IPv6
- A

- B
- IPv6

- IPv4

- IPv4

- E
- IPv6

- F
- IPv6

- D

**Logical view:**

- S
- IPv6

- B
- IPv6

- tunnel

- E
- IPv6

- F
- IPv6

- D

**Data**

IPv4 From:B to E  IPv6 From:S to D
Forwarding Protocols: Review

1. IPv4 uses 32 bit addresses consisting of subnet + host
2. Private addresses can be reused
   \[\Rightarrow\] Helped solve the address shortage to a great extent
3. DHCP is used to automatically allocate addresses to hosts
4. IPv6 uses 128 bit addresses. Requires dual stack or tunneling to coexist with IPv4.

Ref: Read Section 4.3 of the textbook. Try R17 through R29.
Washington University in St. Louis
Generalized Forwarding and SDN

- Planes of Networking
- Data vs. Control Logic
- OpenFlow Protocol
Planes of Networking

- **Data Plane**: All activities involving as well as resulting from data packets sent by the end user, e.g.,
  - Forwarding
  - Fragmentation and reassembly
  - Replication for multicasting

- **Control Plane**: All activities that are necessary to perform data plane activities but do not involve end-user data packets
  - Making routing tables
  - Setting packet handling policies (e.g., security)
  - Base station beacons announcing availability of services

Planes of Networking (Cont)

- **Management Plane**: All activities related to provisioning and monitoring of the networks
  - Fault, Configuration, Accounting, Performance and Security (**FCAPS**).
  - Instantiate new devices and protocols (Turn devices on/off)
  - **Optional** ⇒ May be handled manually for small networks.

- **Services Plane**: Middlebox services to improve performance or security, e.g.,
  - Load Balancers, Proxy Service, Intrusion Detection, Firewalls, SSL Off-loaders
  - **Optional** ⇒ Not required for small networks
Data vs. Control Logic

- Data plane runs at line rate, e.g., 100 Gbps for 100 Gbps Ethernet ⇒ Fast Path
  ⇒ Typically implemented using special hardware, e.g., Ternary Content Addressable Memories (TCAMs)
- Some exceptional data plane activities are handled by the CPU in the switch ⇒ Slow path
  e.g., Broadcast, Unknown, and Multicast (BUM) traffic
- All control activities are generally handled by CPU

Control Logic

Data Logic
OpenFlow: Key Ideas

1. Separation of control and data planes
2. Centralization of control
3. Flow based control

Control logic is moved to a controller
Switches only have forwarding elements
One expensive controller with a lot of cheap switches
OpenFlow is the protocol to send/receive forwarding rules from controller to switches
OpenFlow V1.0

- On packet arrival, match the header fields with flow entries in a table, if any entry matches, perform indicated actions, and update the counters indicated in that entry.

Flow Table:

<table>
<thead>
<tr>
<th>Header Fields</th>
<th>Actions</th>
<th>Counters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Header Fields</td>
<td>Actions</td>
<td>Counters</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>Header Fields</td>
<td>Actions</td>
<td>Counters</td>
</tr>
</tbody>
</table>

Ref: [http://archive.openflow.org/documents/openflow-spec-v1.0.0.pdf](http://archive.openflow.org/documents/openflow-spec-v1.0.0.pdf)

Washington University in St. Louis

©2021 Raj Jain
### Flow Table Example

<table>
<thead>
<tr>
<th>Port</th>
<th>Src MAC</th>
<th>Dst MAC</th>
<th>VLAN ID</th>
<th>Priority</th>
<th>EtherType</th>
<th>Src IP</th>
<th>Dst IP</th>
<th>IP Proto</th>
<th>IP ToS</th>
<th>Src L4 Port</th>
<th>Dst L4 Port</th>
<th>ICMP Type</th>
<th>ICMP Code</th>
<th>Action</th>
<th>Counter</th>
</tr>
</thead>
<tbody>
<tr>
<td>Port 1</td>
<td>* *</td>
<td>0A:C8:*</td>
<td>* * * * *</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>Port 1</td>
<td>102</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Port 2</td>
<td>* *</td>
<td>*</td>
<td>* * *</td>
<td>*</td>
<td>192.168.*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>21</td>
<td>21</td>
<td>Drop</td>
<td>420</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Local</td>
<td>* *</td>
<td>*</td>
<td>* * *</td>
<td>*</td>
<td>0x806</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>*</td>
<td>0x1</td>
<td>*</td>
<td>*</td>
<td>Controller</td>
<td>444</td>
<td></td>
</tr>
</tbody>
</table>

- **Idle timeout**: Remove entry if no packets received for this time
- **Hard timeout**: Remove entry after this time
- **If both are set**, the entry is removed if either one expires.

---


©2021 Raj Jain
Matching

- Set Input Port
- Ether Src
- Ether Dst
- Ether Type
- Set all others to zero

- EtherType = 0x8100?
  - Y: Set VLAN ID
  - N: EtherType = 0x0806?
    - Y: Set VLAN ID
    - N: EtherType = 0x0800?
      - Y: Set VLAN ID
      - N: EtherType = 0x8100?
        - Y: Set VLAN ID
        - N: EtherType = 0x0806?
          - Y: Set VLAN ID
          - N: EtherType = 0x0800?
            - Y: Not IP Fragment?
            - N: Packet lookup using assigned header fields

- EtherType = 0x0806?
  - Y: Set IP Src, IP Dst
  - N: EtherType = 0x0800?
    - Y: Set IP Src, IP Dst
    - N: EtherType = 0x8100?
      - Y: Set VLAN ID
      - N: EtherType = 0x0806?
        - Y: Set VLAN ID
        - N: EtherType = 0x0800?
          - Y: Not IP Fragment?
          - N: Packet lookup using assigned header fields

- EtherType = 0x8100?
  - Y: Set VLAN ID
  - N: EtherType = 0x0806?
    - Y: Set VLAN ID
    - N: EtherType = 0x0800?
      - Y: Not IP Fragment?
      - N: Packet lookup using assigned header fields

- Set VLAN ID
- Set VLAN Priority
- Use EtherType in VLAN tag for next EtherType Check

- IP Proto = 6 or 7
  - Y: Not IP Fragment?
  - N: IP Proto = 1?
    - Y: Use ICMP Type and code for L4 Fields
    - N: Use ICMP Type and code for L4 Fields

- Packet lookup using assigned header fields

- Apply Actions
  - Match Table 0?
    - Y: Packet lookup using assigned header fields
    - N: Apply Actions
  - Match Table n?
    - Y: Packet lookup using assigned header fields
    - N: Apply Actions

- Send to Controller

- Set Src Port, Dst Port for L4 fields
- Use ICMP Type and code for L4 Fields
# Counters

<table>
<thead>
<tr>
<th>Per Table</th>
<th>Per Flow</th>
<th>Per Port</th>
<th>Per Queue</th>
</tr>
</thead>
<tbody>
<tr>
<td>Active Entries</td>
<td>Received Packets</td>
<td>Received Packets</td>
<td>Transmit Packets</td>
</tr>
<tr>
<td>Packet Lookups</td>
<td>Received Bytes</td>
<td>Transmitted Packets</td>
<td>Transmit Bytes</td>
</tr>
<tr>
<td>Packet Matches</td>
<td>Duration (Secs)</td>
<td>Received Bytes</td>
<td>Transmit overrun errors</td>
</tr>
<tr>
<td></td>
<td>Duration (nanosecs)</td>
<td>Transmitted Bytes</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Receive Drops</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Transmit Drops</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Receive Errors</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Transmit Errors</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Receive Frame Alignment Errors</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Receive Overrun errors</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Receive CRC Errors</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Collisions</td>
<td></td>
</tr>
</tbody>
</table>
Actions

- Forward to Physical/Virtual Port $i$
- Enqueue: To a particular queue in the port $\Rightarrow$ QoS
- Drop
- Modify Field: E.g., add/remove VLAN tags, ToS bits, Change TTL
- Masking allows matching only selected fields, e.g., Dest. IP, Dest. MAC, etc.
- If header matches an entry, corresponding actions are performed and counters are updated
- If no header match, the packet is queued and the header is sent to the controller, which sends a new rule. Subsequent packets of the flow are handled by this rule.
- Secure Channel: Between controller and the switch using TLS
Actions (Cont)

- Modern switches already implement flow tables, typically using Ternary Content Addressable Memories (TCAMs)
- Controller can change the forwarding rules if a client moves ⇒ Packets for mobile clients are forwarded correctly
- Controller can send flow table entries beforehand (Proactive) or Send on demand (Reactive). OpenFlow allows both models.
1. **Data plane** consists of packets sent by the users
2. OpenFlow separates data plane from the **control plane** and centralizes the control plane
3. The **controller** makes rules for forwarding and sends to switches
4. Switches match the rules and take specified actions
Network Layer Data Plane: Summary

1. **Forwarding** consists of matching the destination address to a list of entries in a table. **Routing** consists of making that table.
2. IP is a forwarding protocol. IPv4 uses 32 bit addresses in dot-decimal notation. IPv6 uses 128 bit addresses in Hex-Colon notation.
3. **DHCP** is used to assign addresses dynamically.
4. **Private addresses** are used inside an enterprise network. **NAT** allows a single public address to be used by many internal hosts with private addresses.
5. **OpenFlow** separates data plane from control plane and centralizes the control plane.
Acronyms

- ACK  Acknowledgement
- ACM  Automatic Computing Machinery
- AQM  Active Queue Management
- ARP  Address Resolution Protocol
- ATM  Asynchronous Transfer Mode
- BGP  Border Gateway Protocol
- BUM  Broadcast, Unknown, and Multicast
- CAMs Content Addressable Memories
- CBR  Constant bit rate
- CCR  Computer Communications Review
- CIDR Classless Inter-Domain Routing
- CPU  Central Processing Unit
- DHCP Dynamic Host Control Protocol
- DNS  Domain Name Service
- FCAPS  Fault, Configuration, Accounting, Performance and Security
- FCFS  First Come First Served
Acronyms (Cont)

- FTP File Transfer Protocol
- GFR Guaranteed Frame Rate
- HTTP Hyper-Text Transfer Protocol
- ICMP IP Control Message Protocol
- ID Identifier
- IP Inter-Network Protocol
- IPv4 IP Version 4
- IPv6 IP Version 6
- ISP Internet Service Provider
- KISS Keep it simple stupid
- LAN Local Area Network
- MAC Media Access Control
- MS Microsoft
- MTU Maximum Transmission Unit
- NAT Network Address Translation
- PBX Private Branch Exchange
Acronyms (Cont)

- PHY  Physical Layer
- QoS  Quality of Service
- RED  Random Early Drop
- RFC  Request for Comment
- RIP  Routing Information Protocol
- RTT  Round Trip Time
- SDN  Software Defined Networking
- SMTP Simple Mail Transfer Protocol
- SSL  Secure Socket Layer
- TCAM Ternary Content Addressable Memory
- TCP  Transmission Control Protocol
- TLS  Transport Level Security
- ToS  Type of Service
- TTL  Time to live
- UBR  Unspecified bit rate
- UPnP Universal Plug and Play
Acronyms (Cont)

- VBR Variable bit rate
- VCI Virtual Circuit Identifiers
- VLAN Virtual Local Area Network
- VPN Virtual Private Network
- WAN Wide Area Network
- WiFi Wireless Fidelity
Related Modules

CSE 567: The Art of Computer Systems Performance Analysis
https://www.youtube.com/playlist?list=PLjGG94etKypJEKjNAa1n_1X0bWWNyZcof

CSE473S: Introduction to Computer Networks (Fall 2011),
https://www.youtube.com/playlist?list=PLjGG94etKypJWOSPMh8Azcgy5e_10TiDw

CSE 570: Recent Advances in Networking (Spring 2013)
https://www.youtube.com/playlist?list=PLjGG94etKypLHyBN8mOgwJLHD2FFIMGq5

CSE571S: Network Security (Spring 2011),
https://www.youtube.com/playlist?list=PLjGG94etKypKvzfVtutHcPFJXumyyg93u

Video Podcasts of Prof. Raj Jain's Lectures,
https://www.youtube.com/channel/UCN4-5wzNP9-ruOzQMs-8NUw

Student Questions