CSCI-1680
Network Layer:
IP Forwarding realities

Nick DeMarinis

Based partly on lecture notes by Rodrigo Fonseca, David Mazières, Phil Levis, John Jannotti
Administrivia

• Sign up for IP milestone meetings, preferably with your mentor TA, on or before Friday (Oct 6)
  – You don’t need to show an implementation, but you are expected to talk about your design
  – Look for calendar link in email

• IP gearup II: Thursday 5-7pm in CIT368
  – Implementation and debugging tips

• HW1: Due Thursday (HW2 out either Thursday or next Tues)
Today

“Wrinkles” in IP forwarding
• Longest Prefix Match
• IP<->Link layer (ARP, DHCP)
• Network Address Translation (NAT)
• IPv6

After this: Routing
<table>
<thead>
<tr>
<th>Prefix</th>
<th>IF/Next hop</th>
</tr>
</thead>
<tbody>
<tr>
<td>82.14.0.0/16</td>
<td>(A)</td>
</tr>
<tr>
<td>1.3.0.0/16</td>
<td>(B)</td>
</tr>
<tr>
<td>1.3.4.0/24</td>
<td>(C)</td>
</tr>
<tr>
<td>5.6.128.0/20</td>
<td>(D)</td>
</tr>
<tr>
<td>0.0.0.0/0</td>
<td>(Default)</td>
</tr>
</tbody>
</table>

Warmup: based on the table, where would the router send packets destined for the following addresses:

1. 5.6.128.100
2. 1.3.1.1
3. 8.8.8.8
Network A: 82.14.0.0/16
B: 1.3.0.0/16
D: 5.6.128.0/20
C: 1.3.25.0/24

<table>
<thead>
<tr>
<th>Prefix</th>
<th>IF/Next hop</th>
</tr>
</thead>
<tbody>
<tr>
<td>82.14.0.0/16</td>
<td>(A)</td>
</tr>
<tr>
<td>1.3.0.0/16</td>
<td>(B)</td>
</tr>
<tr>
<td>1.3.4.0/24</td>
<td>(C)</td>
</tr>
<tr>
<td>5.6.128.0/20</td>
<td>(D)</td>
</tr>
<tr>
<td>0.0.0.0/0</td>
<td>(Default)</td>
</tr>
</tbody>
</table>

Warmup: based on the table, where would the router send packets destined for the following addresses:
1. 5.6.128.100 - D
2. 1.3.1.1 - B
3. 8.8.8.8 - DEFAULT
4. 1.3.4.8 - X
What happens when prefixes overlap?

An IP can match on more than one row

=> need to pick the most specific (longest) prefix

<table>
<thead>
<tr>
<th>Prefix</th>
<th>IF/Next hop</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.3.0.0/16</td>
<td>(B)</td>
</tr>
<tr>
<td>1.3.4.0/24</td>
<td>(C)</td>
</tr>
<tr>
<td>1.3.4.5/32</td>
<td></td>
</tr>
<tr>
<td>0.0.0.0/0</td>
<td>(Default)</td>
</tr>
</tbody>
</table>

00000001 00000011 xxxxxxxxx xxxxxxxxx

1.3.0.0/16

00000001 00000011 00000100 xxxxxxxxx

More specific => best match!

00000001 00000011 00000100 00000101

1.3.4.0/24

Other examples you’ll see...

00000001 00000011 xxxxxxxxx xxxxxxxxx xxxxxxxxx xxxxxxxxx

0.0.0.0/0

=> Least specific!
(Used for default “catchall” routes)

00000001 00000011 00000100 00000101 00000101

1.2.3.5/32

=> Most specific!
(Refers to a single host, often a local IP)

00000001 00000011 00000100 00000101 00000101

=Longest prefix matching: can keep forwarding tables small by summarizing routes where possible, otherwise using specific prefixes
What happens at the link layer?

What does it mean to send to IF1?

<table>
<thead>
<tr>
<th>Prefix</th>
<th>IF/Next hop</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.2.1.0/24</td>
<td>IF1</td>
</tr>
<tr>
<td>1.2.2.0/24</td>
<td>IF2</td>
</tr>
<tr>
<td>8.0.0.0/30</td>
<td>IF0</td>
</tr>
<tr>
<td>Default</td>
<td>8.0.0.2</td>
</tr>
</tbody>
</table>
"Local delivery": what does it mean to send to IF1?

So far: “easy” to communicate with nodes on the same network. But how?

In order to send on local net, need:
- Dest IP (L3)
- Dest MAC address

<table>
<thead>
<tr>
<th>Link</th>
<th>IP</th>
<th>10.2.4.100</th>
<th>Dest</th>
<th>1.2.1.3</th>
</tr>
</thead>
</table>

Local delivery:
what does it mean to send to IF1?
“Glue” between L2 and L3

Need a way to connect get link layer info (mac address) from network-layer info (IP address)

“What MAC address has IP 1.2.3.4?”
“Glue” between L2 and L3

Need a way to connect get link layer info (mac address) from network-layer info (IP address)

“What MAC address has IP 1.2.3.4?”

Ask the network!
=> Address Resolution Protocol (ARP)
ARP: Address resolution protocol

Given an IP address, ask network for the MAC address

- Maps IP addresses to mac addresses
  - Request: “Who has 1.2.3.4?”
  - Response: “aa:bb:cc:dd:ee:ff is at 1.2.3.4”
ARP: Address resolution protocol

Given an IP address, ask network for the MAC address

- Maps IP addresses to mac addresses
  - Request: “Who has 1.2.3.4?”
  - Response: “aa:bb:cc:dd:ee:ff is at 1.2.3.4”

- ARP table: hosts cache IP->mac mappings

- Requests send to broadcast address: ff:ff:ff:ff:ff:ff:ff
  - Anyone can respond: problem?
PING 1.2.3.4

WHO HAS 1.2.3.4?

ETH | SRC | DST
| A:A:A | ff:ff:ff:ff:ff

1.2.3.4 IS AT B:B:B:B
Responses are cached at the host in the ARP table:

Maps IP => MAC address

Then when you send the next packet, check the ARP table for the MAC address
If table miss, send an ARP request
Who has 1.2.1.3?

Request is sent to broadcast address! Anyone can respond. Problem?
Example

```
# arp -n

<table>
<thead>
<tr>
<th>Address</th>
<th>HWtype</th>
<th>HWaddress</th>
<th>Flags</th>
<th>Mask</th>
<th>Iface</th>
</tr>
</thead>
<tbody>
<tr>
<td>172.17.44.1</td>
<td>ether</td>
<td>00:12:80:01:34:55</td>
<td>C</td>
<td></td>
<td>eth0</td>
</tr>
<tr>
<td>172.17.44.25</td>
<td>ether</td>
<td>10:dd:b1:89:d5:f3</td>
<td>C</td>
<td></td>
<td>eth0</td>
</tr>
<tr>
<td>172.17.44.6</td>
<td>ether</td>
<td>b8:27:eb:55:c3:45</td>
<td>C</td>
<td></td>
<td>eth0</td>
</tr>
<tr>
<td>172.17.44.5</td>
<td>ether</td>
<td>00:1b:21:22:e0:22</td>
<td>C</td>
<td></td>
<td>eth0</td>
</tr>
</tbody>
</table>
```

All entries should have timeout, etc.
Subnet “A”: 1.2.1.0/24

Subnet “B”: 1.2.2.0/24
From 1.2.1.2 → 1.2.2.100

1. ETH | SRC: A:A:A | DST: 1.2.2.108
   | IP: 1.2.1.2 | MAC address of IF1

2. ETH | SRC: 1F2 | DST: B:B:B
   | IP: 1.2.2.2 | 1.2.2.100
   | D: 8.8.8.8 | Change at each cross link
   | E: does not respond | End DST of packet
   | B: not checked by default | NOT CHECKED BY DEFAULT!

3. ETH | SRC: B:B:B | DST: 1F2
   | IP: 1.2.2.100 | 1.2.1.2
   | ✖ SRC ADDN IS USED FOR RESPONSE.
How do you get an IP address?
Getting an IP

Two ways to configure an IP for a host:

• **Static** configuration: manually specify IP address, mask, gateway, ...

  => More common with network devices that don’t change often

• **Automatic**: ask the network for an IP when you connect!

  => Most common for end hosts
  => Dynamic Host Configuration Protocol (DHCP)
Host A

Src: A's MAC address
Dst: ff:ff:ff:ff:ff:ff
DHCPDISCOVER

DHCP server

AT START, DON'T KNOW SERVER'S IP!
Host A

**Src:** A's MAC address
**Dst:** ff:ff:ff:ff:ff:ff
**DHCPDISCOVER**

**Src:** <Server MAC address>
**Dst:** ff:ff:ff:ff:ff:Ff
**DHCPOFFER:**
- **Your IP:** 192.168.1.102
- **Mask:** 255.255.255.0
- **Router:** 192.168.1.1

... (More steps after this)

DHCP server

“Enough to set up host to use net...”

“**Multiple servers for redundancy**...”
Host A

Src: A's MAC address
Dst: ff:ff:ff:ff:ff:ff
DHCPDISCOVER

(DHCP server)

Src: <Server MAC address>
Dst: ff:ff:ff:ff:ff:Ff
DHCPOFFER:
Your IP: 192.168.1.102
Mask: 255.255.255.0
Router: 192.168.1.1
...

(More steps after this)

=> Again, host needs to use broadcast address. Why?
=> Problem?
A home router

What’s in this thing?

- OS (Linux)
- IP forwarding
- DHCP
- Ethernet switch

Inside:
- Local network
- 192.168.1.0/24

Outside:
- Internet
- DMZ
Story time
About those home routers…

You get just one IP from your ISP…

=> Need to share IP among many devices on the same network!
About those home routers…

You get just one IP from your ISP…

=> Need to share IP among many devices on the same network!

Common to create a “private” IP range used within local network

=> Routers need to do extra work to share public IP among private IPs

=> **Network Address Translation (NAT)**

(A form of connection multiplexing)
Private IPs (RFC1918)

Some IP ranges are reserved:

<table>
<thead>
<tr>
<th>Prefix</th>
<th>Use</th>
</tr>
</thead>
<tbody>
<tr>
<td>127.0.0.0/8</td>
<td>“Loopback” address—always for current host</td>
</tr>
<tr>
<td>10.0.0.0/8</td>
<td>USEDFOR INTERNAL STUFF</td>
</tr>
<tr>
<td>192.168.0.0/16</td>
<td>Reserved for private internal networks (RFC1918)</td>
</tr>
<tr>
<td>172.16.0.0/12</td>
<td>HOME NETWORKS - DOCKEN</td>
</tr>
</tbody>
</table>

- Many networks will use these blocks internally
Network Address Translation

- What happens when hosts need to share an IP address?
- How to map private IP space to public IPs?
Network Address Translation (NAT)

- Despite CIDR, it’s still difficult to allocate addresses ($2^{32}$ is only 4 billion)
- NAT “hides” entire network behind one address
- Hosts are given private addresses
- Routers map outgoing packets to a free address/port
- Router reverse maps incoming packets
- Problems?
How NAT Works (in General)

1. Packet from A
2. Router Translator

Router Stores:
10.0.0.1:5000 \rightarrow 5.6.2.5:5000

Response from S

1.2.3.4:80
10.0.0.10:5000
Router uses port numbers to "multiplex" connections to one IP.

End to end connectivity is broken!

Outside host can't connect unless inside host started a connection.

NAT vs. SnoeCAST

```
Control     Listener
     :10000
```

Hello, connect to me on port 10000

FTP    - VoIP    - Games
NAT Traversal

Various methods, depending on the type of NAT

Examples:

- **ICE**: Interactive Connectivity Establishment (RFC8445)
- **STUN**: Session Traversal Utilities for NAT (RFC5389)

One idea: connect to external server via UDP, it tells you the address/port