CSCI1680
Network Layer: IP & Forwarding II

Nick DeMarinis

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

- **IP Project**
  - You should get an email today with your group assignment and a repo link
  - Milestone: meet with me/a TA on/before October 11 to discuss your design (signups soon)

- **Look for a post soon about debugging/workflow**

- **HW2: Announcement soon**
  - Stuff we’ve covered + warmup for IP!
Today

Start of network layer

• IP forwarding mechanics
• About the IP project
Recall: IP Addressing

What’s in an IP address?
138.16.161.209
Recall: IP addressing

A typical IP configuration looks like this:
Identifying the network

• Prefix or Subnet mask => identifies what part of the address is the network part

Why do we care?
Routers forward to networks, not individual hosts!
How do we move packets between networks?
IP forwarding

Given a packet, decide where to send it

<table>
<thead>
<tr>
<th>Prefix</th>
<th>Interface/Next hop</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>
A forwarding table (my laptop)

deeemer@ceres ~ % ip route default via 10.3.128.1 dev wlp2s0
10.3.128.0/18 dev wlp2s0 proto dhcp scope link src 10.3.135.44 metric 3003
172.18.0.0/16 dev docker0 proto kernel scope link src 172.18.0.1
192.168.1.0/24 dev enp0s31f6 proto kernel scope link src 192.168.1.1
Forwarding: examples

<table>
<thead>
<tr>
<th>Prefix</th>
<th>Interface/Next hop</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Routing based on networks
A routing table

R6#sh ip ro
Gateway of last resort is 108.34.215.1 to network 0.0.0.0

S* 0.0.0.0/0 [1/0] via 108.34.215.1
    10.0.0.0/8 is variably subnetted, 7 subnets, 3 masks
C  10.1.0.0/24 is directly connected, wlan-ap0
L  10.1.0.2/32 is directly connected, wlan-ap0
O IA 10.1.44.1/32 [110/1001] via 10.20.30.33, 3w4d, Tunnel0
C  10.1.48.0/24 is directly connected, Loopback0
L  10.1.48.1/32 is directly connected, Loopback0
C  10.20.30.32/31 is directly connected, Tunnel0
L  10.20.30.32/32 is directly connected, Tunnel0
108.0.0.0/8 is variably subnetted, 2 subnets, 2 masks
C  108.34.215.0/24 is directly connected, GigabitEthernet0/0
L  108.34.215.208/32 is directly connected, GigabitEthernet0/0
172.16.0.0/16 is variably subnetted, 2 subnets, 2 masks
C  172.16.98.0/24 is directly connected, Vlan98
L  172.16.98.1/32 is directly connected, Vlan98
172.17.0.0/16 is variably subnetted, 6 subnets, 3 masks
O IA 172.17.44.0/24 [110/1001] via 10.20.30.33, 3w4d, Tunnel0
C  172.17.48.0/24 is directly connected, Vlan20
L  172.17.48.1/32 is directly connected, Vlan20
C  172.17.49.0/25 is directly connected, Vlan50
A routing table

R6#sh ip ro
Codes: L - local, C - connected, S - static, R - RIP, M - mobile, B - BGP
    D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
    N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2
    E1 - OSPF external type 1, E2 - OSPF external type 2
    i - IS-IS, su - IS-IS summary, L1 - IS-IS level-1, L2 - IS-IS level-2
    ia - IS-IS inter area, * - candidate default, U - per-user static route
    o - ODR, P - periodic downloaded static route, H - NHRP, l - LISP
    + - replicated route, % - next hop override

Gateway of last resort is 108.34.215.1 to network 0.0.0.0

S*  0.0.0.0/0 [1/0] via 108.34.215.1
     10.0.0.0/8 is variably subnetted, 7 subnets, 3 masks
C   10.1.0.0/24 is directly connected, wlan-ap0
L   10.1.0.2/32 is directly connected, wlan-ap0
O IA 10.1.44.1/32 [110/1001] via 10.20.30.33, 3w4d, Tunnel0
C   10.1.48.0/24 is directly connected, Loopback0
L   10.1.48.1/32 is directly connected, Loopback0
C   10.20.30.32/31 is directly connected, Tunnel0
L   10.20.30.32/32 is directly connected, Tunnel0
### show route table inet.0 active-path

*inet.0: 866991 destinations, 13870153 routes (866991 active, 0 holddown, 0 hidden)*
+ = Active Route, - = Last Active, *= Both

<table>
<thead>
<tr>
<th>Network</th>
<th>Route Details</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.0.0.0/0</td>
<td>*[Static/5] 5w0d 19:43:09 &gt; to 12.0.1.1 via em0.0</td>
</tr>
<tr>
<td>1.0.0.0/24</td>
<td>*[BGP/170] 1d 10:24:47, localpref 100, from 12.122.83.238, AS path: 7018 3356 13335 I, validation-state: valid &gt; to 12.0.1.1 via em0.0</td>
</tr>
<tr>
<td>1.0.4.0/22</td>
<td>*[BGP/170] 1d 10:24:47, localpref 100, from 12.122.83.238, AS path: 7018 3356 4826 38803 I, validation-state: valid &gt; to 12.0.1.1 via em0.0</td>
</tr>
<tr>
<td>1.0.4.0/24</td>
<td>*[BGP/170] 1d 10:24:47, localpref 100, from 12.122.83.238, AS path: 7018 3356 4826 38803 I, validation-state: valid &gt; to 12.0.1.1 via em0.0</td>
</tr>
<tr>
<td>1.0.5.0/24</td>
<td>*[BGP/170] 1d 10:24:47, localpref 100, from 12.122.83.238, AS path: 7018 3356 4826 38803 I, validation-state: valid &gt; to 12.0.1.1 via em0.0</td>
</tr>
<tr>
<td>1.0.6.0/24</td>
<td>*[BGP/170] 1d 10:24:47, localpref 100, from 12.122.83.238, AS path: 7018 3356 4826 38803 I, validation-state: valid &gt; to 12.0.1.1 via em0.0</td>
</tr>
</tbody>
</table>
How does forwarding actually work?
The IPv4 Header

Defined by RFC 791
RFC (Request for Comment): defines network standard
Most Important fields

• **Version**: 4 for IPv4 packets, 6 for IPv6
• **Source address**: where the packet came from
• **Destination address**: where the packet is going

(continued…)

17
More important fields

• **TTL (time-to-live):** decremented each hop
  – Can prevent forwarding loops (and do other stuff…)

• **Checksum:** computed over **header** (very weak!)

• **Protocol identifier:** describes what’s in the packet
  – 6: TCP, 17: UDP, 1: ICMP, …
  – Defines the type of the payload
Less important fields

• Header length: in 32-bit units
  – >5 implies use of IP options
  – Almost all routers ignore IP options

• Fragmentation
  – Network can fragment a packet if next link requires a small frame
  – Most routers don’t fragment (or reassemble fragments)

• We won’t talk about…
  – Type of Service (TOS): basic traffic classification
  – Identifier: might have special meaning on some networks
Forwarding steps

What does a router do when it receives a packet?
Forwarding mechanics

When an IP packet arrives at a host/router:

• Is it valid? Verify checksum over header

• Is it for me? If dest IP == your address, send to OS

• If not, where should it go?
  – Consult forwarding table => find next hop
  – Decrement TTL
  – Send packet to next hop
Traceroute

• When TTL reaches 0, router may send back an error
  – ICMP TTL exceeded
• If it does, we can identify a path used by a packet!
Traceroute example

[deemer@Warsprite ~]$ traceroute -q 1 google.com
traceroute to google.com (142.251.40.174), 30 hops max, 60 byte packets
  1 router1-nac.linode.com (207.99.1.13)  0.621 ms
  2 if-0-1-0-0-0.gw1.cjj1.us.linode.com (173.255.239.26)  0.499 ms
  3 72.14.222.136 (72.14.222.136)  0.949 ms
  4 72.14.222.136 (72.14.222.136)  0.919 ms
  5 108.170.248.65 (108.170.248.65)  1.842 ms
  6 lga25s81-in-f14.1e100.net (142.251.40.174)  1.812 ms
Traceroute example

[deemer@Warsprite ~]$ traceroute -q 1 amazon.co.uk
traceroute to amazon.co.uk (178.236.7.220), 30 hops max, 60 byte packets
  1  router2-nac.linode.com (207.99.1.14) 0.577 ms
  2  if-11-1-0-1-0.gw2.cjj1.us.linode.com (173.255.239.16) 0.461 ms
  3  ix-et-2-0-2-0.tcore3.njy-newark.as6453.net (66.198.70.104) 1.025 ms
  4  be3294.ccr41.jfk02.atlas.cogentco.com (154.54.47.217) 2.938 ms
  5  be2317.ccr41.lon13.atlas.cogentco.com (154.54.30.186) 69.725 ms
  6  be2350.rcr21.b023101-0.lon13.atlas.cogentco.com (130.117.51.138) 69.947 ms
  7  a100-row.demarc.cogentco.com (149.11.173.122) 71.639 ms
  8  150.222.15.28 (150.222.15.28) 78.217 ms
  9  150.222.15.21 (150.222.15.21) 84.383 ms
 10   *
 11  150.222.15.4 (150.222.15.4) 74.529 ms
     ...  
Demo: IP project
Coming up…

- ARP: Mapping IPs to MAC addresses
- How are addresses assigned?
- NAT: When it gets complicated
- Routing algorithms: how to build forwarding tables

Fill out the group preference survey for the IP project (announcement soon) by tomorrow (Sep 30) by 11:59PM
How to avoid loops?

<table>
<thead>
<tr>
<th>Version</th>
<th>IHL</th>
<th>TOS</th>
<th>Total length</th>
</tr>
</thead>
<tbody>
<tr>
<td>Identification</td>
<td>Flags</td>
<td>Fragment offset</td>
<td></td>
</tr>
<tr>
<td>TTL</td>
<td>Protocol</td>
<td>Header checksum</td>
<td></td>
</tr>
</tbody>
</table>

**TTL (Time to Live):** Decrement by 1 at each hop, send back error at 0

**traceroute:** tool to send packets with increasing TTLs

=> can learn about network paths!
Backup slides from last lecture
<table>
<thead>
<tr>
<th>Prefix</th>
<th>Binary Representation</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.2.0.0/16</td>
<td>00000001 00000010 x x x x x x x x x x x x</td>
</tr>
<tr>
<td>8.0.0.0/8</td>
<td>00001000 x x x x x x x x x x x x x x x x</td>
</tr>
<tr>
<td>123.10.1.0/24</td>
<td>01111011 00001010 00000001 x x x x x x x x</td>
</tr>
<tr>
<td>201.112.10.200/30</td>
<td>11001001 01110000 00001010 110010xx</td>
</tr>
</tbody>
</table>
How IP forwarding works

Assume:

- Communicating on same network is easy—this is the link-layer’s job!
- Can map IP addresses to MAC addresses (more on this later)

How to reach an address outside this network?

Send packets to a router, which forwards IP packets to other networks
Forwarding IP packets

Src: 1.2.1.3
Dst: 1.2.2.100
. . .

To more networks (ie, Internet)

1.2.1.3
1.2.1.200

1.2.1.1
1.2.2.1

IF0

IF1

IF2

1.2.2.100

1.2.2.105
Forwarding IP packets

Src: 1.2.1.3
Dst: 1.2.2.100

To more networks (i.e., Internet)
Forwarding IP packets

Prefix: 1.2.1.2 1.2.1.3 1.2.1.200
Interface/Next hop: IF1 IF0 IF2

Source (Src): 1.2.1.3
Destination (Dst): 1.2.2.100

To more networks (ie, Internet)

Prefix: 1.2.2.100 1.2.2.105
Forwarding IP packets

- Source: 1.2.1.3
- Destination: 1.2.2.100
- To more networks (i.e., Internet)

Prefix | Interface
--- | ---
1.2.1.0/24 | IF1
1.2.2.0/24 | IF2
<everything else> | (IF0)
Wait, what happens at the link layer?
What about the rest?

How to reach networks that aren’t directly connected?

<table>
<thead>
<tr>
<th>Prefix</th>
<th>Interface</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>&lt;everything else&gt;</td>
<td>IF0</td>
</tr>
</tbody>
</table>

To more networks (ie, Internet)
What about the rest?

- Need “next hop” IP: another router that knows about other networks
  - How to reach it? Check table again!
- “Default gateway”: where to send to reach anything not in the table

<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>128.148.0.0/16</td>
<td>1.2.1.5</td>
</tr>
<tr>
<td>Default</td>
<td>8.0.0.2</td>
</tr>
</tbody>
</table>
The forwarding table

Exploits hierarchical structure of addresses: know how to reach *networks*, not individual hosts

- Table is keyed is a network prefix, not a whole address
- Select best prefix with *longest prefix matching* (more on this later)

<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>128.148.0.0/16</td>
<td>1.2.1.5</td>
</tr>
<tr>
<td>Default</td>
<td>8.0.0.2</td>
</tr>
</tbody>
</table>
How many addresses are in the network 1.1.0.0/20?
Putting it all together...

- The more connected a router becomes, the more complex its forwarding table... and the more it may change!

- **Routing algorithms**: routers exchange path information to their forwarding tables (more on this later)
Goal: find the most specific (i.e., longest) prefix matching the destination

How to reach 1.2.2.100?

<table>
<thead>
<tr>
<th>Prefix</th>
<th>Interface</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>0.0.0.0/0</td>
<td>IF0</td>
</tr>
</tbody>
</table>

Output: IF2

Longest Prefix Matching (LPM): can represent entire IP space in (small) table!
Some ISP

<table>
<thead>
<tr>
<th>Prefix</th>
<th>Interface</th>
</tr>
</thead>
<tbody>
<tr>
<td>128.148.0.0/16</td>
<td>IF1</td>
</tr>
<tr>
<td>1.3.0.0/16</td>
<td>IF2</td>
</tr>
<tr>
<td>5.6.128.0/20</td>
<td>IF3</td>
</tr>
<tr>
<td>128.148.100.0/24</td>
<td>IF4</td>
</tr>
<tr>
<td>0.0.0.0/0</td>
<td>8.0.0.2</td>
</tr>
</tbody>
</table>

Brown: 128.148.0.0/16
Customer 2: 1.3.0.0/16
Customer 3: 5.6.128.0/20
Brown': 128.148.100.0/24

Dst: 128.148.105.207
... 

Dst: 128.148.100.104
...