CSCI1680
Network Layer: IP & Forwarding

Nick DeMarinis

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

- Snowcast: was due last night
- IP Project: Out later today/tomorrow (Sep 30)
  - Fill out group preference form by 11:59pm tomorrow (Sep 30)
- HW2: Announcement soon
  - Stuff we’ve covered + warmup for IP!
Today

Start of network layer

• Network layer: Internet Protocol (IP) (v4)
• Mechanics of IP forwarding
• Intro to IP project
LAST LECTURE'S PLA

H1

S1

port: 1

port: 2

port: 3

port: 4

S2

H2

H4

H3

H5

S1

MAC Address Port
H1 1
H4 4

S2

MAC Address Port
H1 5
Bridges and Extended LANs

• Single Ethernet collision domain has limitations
  – Limits performance, distance, ...
• Next step: separate collision domains with bridges
  – Operates on Ethernet addresses
  – Forwards packets from one collision domain to others
• Modern ethernet uses switches: all hosts directly connected to a bridge
More switching challenges:

- Dealing w/ loops
- Line rate forwarding...
- More later!
Layers, Services, Protocols

- **Physical**
  - Service: move bits to other node across link

- **Link**
  - Service: move frames to other node across link.
  - May add reliability, medium access control

- **Network**
  - Service: move packets to any other node in the network
  - Internet Protocol (IP)

- **Transport**
  - Service: multiplexing applications
  - Reliable byte stream to other node (TCP),
  - Unreliable datagram (UDP)

- **Application**
  - Service: user-facing application.
  - Application-defined messages
Internet Protocol (IP) Goals

How to connect everyone?

• Glue lower-level networks together
• A network of networks!
Internet Protocol (IP) Goals

How to connect everyone?

• Glue lower-level networks together
• A network of networks!
• Router: device that forwards packets between networks
Internet Protocol (IP) Goals

How to connect everyone?

• Glue lower-level networks together
• A network of networks!
• Router: device that forwards packets between networks

Doesn’t this sound like switching?
Inter-networking Challenges

• Networks are heterogeneous (eg. Wifi vs. Ethernet)
  – Different frame formats
  – Different service models
  – Different packet sizes/bandwidths
Inter-networking Challenges

• Networks are heterogeneous (eg. Wifi vs. Ethernet)
  – Different frame formats
  – Different service models
  – Different packet sizes/bandwidths

• Scaling
  – Link-layer forwarding strategies don’t scale to Internet!
Map of the Internet, 2021 (via BGP)
OPTE project
A Bit of History

• Packet switched networks: Arpanet’s IMPs
  – Late 1960’s
  – RFC 1, 1969!
  – Segmentation, framing, routing, reliability, reassembly, primitive flow control

• Network Control Program (NCP)
  – Provided connections, flow control
  – Assumed reliable network: IMPs
  – Used by programs like telnet, mail, file transfer
A Bit of History

• Packet switched networks: Arpanet’s IMPs
  – Late 1960’s
  – RFC 1, 1969!
  – Segmentation, framing, routing, reliability, reassembly, primitive flow control

• Network Control Program (NCP)
  – Provided connections, flow control
  – Assumed reliable network: IMPs
  – Used by programs like telnet, mail, file transfer

• Wanted to connect multiple networks
  – Not all reliable, different formats, etc…
THE ARPA NETWORK
DEC 1969
Abb. 4 ARPA NETwork, topologische Karte. Stand Juni 1974.
How would you design such a protocol?

- Circuits or packets?
  - Predictability

Circuit switching:
  - Nodes decide path beforehand

Packet switching:
  - Less state
  - Break messages into packets, each node decides what to do
How would you design such a protocol?

- Circuits or packets?
  - Predictability
- What service model?
  - Reliability, timing, bandwidth guarantees
How would you design such a protocol?

- Circuits or packets?
  - Predictability
- What service model?
  - Reliability, timing, bandwidth guarantees
- How to enable connectivity?
  - How do you find a particular host?
  - How do you get a message there?
  - What happens when a host joins/leaves?
1974: TCP/IP Introduced

- Vint Cerf, Robert Kahn build protocol to replace NCP
- Initial design: single protocol providing a unified reliable pipe
1974: TCP/IP Introduced

- Vint Cerf, Robert Kahn build protocol to replace NCP
- Initial design: single protocol providing a unified reliable pipe
- Different requirements soon emerged, and the two were separated
  - IP: basic datagram service among hosts
  - TCP: reliable transport
  - UDP: unreliable multiplexed datagram service
IP’s Decisions
IP’s Decisions

- **Circuits or packets?** Packets.

  Routes don’t need to remember per connection state.
IP’s Decisions

• Circuits or packets? Packets.

• Service model?
  – Lowest common denominator: best effort, connectionless datagram
IP’s Decisions

- **Circuits or packets?** Packets.
- **Service model?**
  - Lowest common denominator: best effort, connectionless datagram
- **Enabling connectivity?**
  - IP header: common message format
  - IP address: each host has an address, based on hierarchical structure of network
An excellent read

David D. Clark, “The design Philosophy of the DARPA Internet Protocols”, 1988

- Primary goal: multiplexed utilization of existing interconnected networks
- Other goals:
  - Communication continues despite loss of networks or gateways
  - Support a variety of communication services
  - Accommodate a variety of networks
  - Permit distributed management of its resources
  - Be cost effective
  - Low effort for host attachment
  - Resources must be accountable
Internet Protocol

IP runs on all hosts and routers
- Provides addressing: how we name nodes in an IP network
- Provides forwarding: how routers move packets based on the destination address
- Later: routing: how routers build forwarding rules
IP’s Service Model

- **Connectionless** (datagram-based)
  - Best-effort delivery (unreliable service)
    - packets may be lost
    - packets may be delivered out of order
    - duplicate copies of packets may be delivered
    - packets may be delayed for a long time
  - It’s the **lowest common denominator**
    - A network that delivers no packets fits the bill!
    - All these can be dealt with above IP (if probability of delivery is non-zero…)
IP Addressing

IP Version 4: Each address is a 32-bit number:

128.148.16.7

10000000 10010100 00010000 00000111
IP Addressing

IP Version 4: Each address is a 32-bit number: 128.148.16.7

Notation
- Write each byte ("octet") as a decimal number
- This is called "dotted decimal" or "dotted quad" notation

128.148.16.7

10000000 10010100 00010000 00000111

2^22 possible addresses
≈ 4 billion possible IPs
An IP address identifies...

• Who a host is: A unique number
• Where it is on the Internet

HOST #1242

BROWN NETWORK

128.148.16.7
An IP address identifies…

- Who a host is: A unique number
- Where it is on the Internet
- Networks are allocated ranges of IPs by global authority (ICANN)

*ICANN (Internet Corporation for Assigned Names and Numbers)
IP Addressing

An IP address identifies…

• Who a host is: A unique number
• Where it is on the Internet
• Networks are allocated ranges of IPs by global authority (ICANN)
  – Further subdivided by regions, ISPs, …
• Some IPs have special uses (eg. 127.0.0.1)

128.148.16.7

*ICANN (Internet Corporation for Assigned Names and Numbers)
An IP address identifies...

- Who a host is: A unique number
- Where it is on the Internet
- Networks are allocated ranges of IPs by global authority (ICANN)
  - Further subdivided by regions, ISPs, ...
- Some IPs have special uses (eq. 127.0.0.1)


*ICANN (Internet Corporation for Assigned Names and Numbers)
Brown owns the range: 128.148.xxx.xxx

10000000 10010100 xxxxxxxx xxxxxxxx

Network part
Identifies Brown (to the Internet)

Host part
Denotes individual hosts within the Brown Network

/26 Hosts
/2 65,536 Hosts

128.148.0/16
A network can designate IP addresses for its own hosts within its address range.

For 128.148.xxx.xxx:
A network can designate IP addresses for its own hosts within its address range.

For 128.148.xxx.xxx:
A network can designate IP addresses for its own hosts within its address range.

For 128.148.xxx.xxx:

\[
\begin{align*}
&10000000 \quad 10010100 \quad xxxxxxxx \quad xxxxxxxx
\end{align*}
\]
# Common prefixes

<table>
<thead>
<tr>
<th>Prefix</th>
<th>Binary</th>
<th>Decimal</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.2.0.0/16</td>
<td>00000001 00000010 xxxxxxxxxx xxxxxxxxxx</td>
<td></td>
</tr>
<tr>
<td>8.0.0.0/8</td>
<td>00001000 xxxxxxxx xxxxxxxx xxxxxxxx</td>
<td></td>
</tr>
<tr>
<td>123.10.1.0/24</td>
<td>01111011 00001010 00000001 xxxxxxxxxx</td>
<td></td>
</tr>
<tr>
<td>201.112.10.200/30</td>
<td>11001001 01110000 00001010 110010xx</td>
<td></td>
</tr>
</tbody>
</table>

- **1.2.0.0/16**: $2^{24}$ hosts
- **8.0.0.0/8**: $2^{16}$ hosts
- **123.10.1.0/24**: $2^8$ hosts
- **201.112.10.200/30**: $2^2$ hosts
Example

How many addresses are in the network 1.1.0.0/20?

- The first 20 bits are the network part.
- The last 12 bits are the host part.
- \(2^{12} = 4096\) possible hosts.
How do we move packets between networks?

⇒ Forwarding.
How IP forwarding works

Assume:

- Communicating on same network is easy—this is the link-layer’s job!
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 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?
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

To more networks (i.e., Internet)

1.2.1.2
1.2.1.3
1.2.1.200

1.2.2.105
1.2.2.100

1.2.2.0/24

1.2.1.0/24

1.2.1.1
IF0

1.2.2.1
IF2

1.2.2.124

IF1
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

IF0

1.2.1.1

IF1

1.2.2.1

IF2

1.2.2.100

1.2.2.105

1.2.1.0/24

NOT FOR LOCAL NETWORK
(DOESN'T START W/ 1.2.1.X)
SO SEND TO ROUTER
Forwarding IP packets

Src: 1.2.1.3
Dst: 1.2.2.100

To more networks (ie, Internet)
Forwarding IP packets

Src: 1.2.1.3
Dst: 1.2.2.100

To more networks (ie, Internet)

Prefix  Interface
1.2.1.0/24  IF1
1.2.2.0/24  IF2
*  IF0
Forwarding IP packets

<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>

Src: 1.2.1.3
Dst: 1.2.2.100

To more networks (i.e., Internet)
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>

(IF0, IF1, IF2)
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>
A forwarding table

# ip route
127.0.0.0/8 via 127.0.0.1 dev lo
172.17.44.0/24 dev enp7s0 proto kernel scope link src 172.17.44.22 metric 204
default via 172.17.44.1 dev eth0 src 172.17.44.22 metric 204
The IPv4 Header

<table>
<thead>
<tr>
<th>0</th>
<th>4</th>
<th>8</th>
<th>16</th>
<th>31 bit</th>
</tr>
</thead>
<tbody>
<tr>
<td>Version</td>
<td>IHL</td>
<td>TOS</td>
<td>Total length</td>
<td></td>
</tr>
<tr>
<td>Identification</td>
<td>Flags</td>
<td>Fragment offset</td>
<td></td>
<td></td>
</tr>
<tr>
<td>TTL</td>
<td>Protocol</td>
<td>Header checksum</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Source address</td>
<td>Destination address</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Options</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Data</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

- 20 bytes
- 0-40 bytes
- Up to 65536 bytes
For us, the "Link Layer" is a UDP socket.