Network Layer
Where we are in the Course

• Moving down to the Network Layer!
Network Layer

- How to deliver packets globally
  - Routing as the primary concern

<table>
<thead>
<tr>
<th>Application</th>
</tr>
</thead>
<tbody>
<tr>
<td>Transport</td>
</tr>
<tr>
<td>Network</td>
</tr>
<tr>
<td>Link</td>
</tr>
<tr>
<td>Physical</td>
</tr>
</tbody>
</table>
Why do we need a Network layer?

• Assume we can send packets between “directly connected” hosts
  • The link layer
  • Based on broadcast

• Broadcast doesn’t scale to the Internet!
Network Layer Approach

• Scaling:
  • Hierarchy, in the form of prefixes

• Heterogeneity:
  • IP for internetworking

• Bandwidth Control:
  • Lowest-cost routing
  • Later QOS (Quality of Service)
Topics

• Network service models
  • Datagrams (packets), virtual circuits
• IP (Internet Protocol)
  • Internetworking
  • Forwarding (Longest Matching Prefix)
  • Helpers: ARP and DHCP
  • Fragmentation and MTU discovery
  • Errors: ICMP (traceroute!)
  • IPv6, scaling IP to the world
  • NAT, and “middleboxes”

• Routing Algorithms
Routing vs. Forwarding

- **Routing** is the process of deciding in which direction to send traffic
  - Network wide (global) and expensive
Routing vs. Forwarding

- **Forwarding** is the process of sending a packet
  - Node process (local) and fast
Networking Services
• What kind of service does the Network layer provide to the Transport layer?
  • How is it implemented at routers?
Two Network Service Models

• Datagrams, or connectionless service
  • Like postal letters
  • (IP as an example)

• Virtual circuits, or connection-oriented service
  • Like a telephone call
Store-and-Forward Packet Switching

• Both models are implemented with store-and-forward packet switching
  • Routers receive a complete packet, storing it temporarily if necessary before forwarding it onwards
  • We use statistical multiplexing to share link bandwidth over time
Store-and-Forward

• Switching element has internal buffering for contention
Store-and-Forward

• Simplified view with per port output buffering
  • Buffer is typically a FIFO (First In First Out) queue
  • If full, packets are discarded (congestion, later)
Datagram Model

• Packets contain a **destination address**; each router uses it to forward packets, maybe on different paths
Datagram Model

- Each router has a forwarding table keyed by destination address
  - Gives next hop for each destination address; may change
IP (Internet Protocol)

- Network layer of the Internet, uses **datagrams**
- IPv4 carries 32 bit addresses on each packet (often 1.5 KB)
Virtual Circuit Model

• Three phases:
  1. Connection establishment, circuit is set up
     • Path is chosen, circuit information stored in routers
  2. Data transfer, circuit is used
     • Packets are forwarded along the path
  3. Connection teardown, circuit is deleted
     • Circuit information is removed from routers

• Just like a(n old fashioned) telephone circuit, but virtual in that no bandwidth need be reserved; statistical sharing of links
Virtual Circuits

- Packets contain a short label to identify the circuit.
- Labels don’t have global meaning, only unique on a link.

ISP’s equipment
Virtual Circuits

• Each router has a **forwarding table** keyed by circuit
  • Gives output line and next label to place on packet
MPLS (Multi-Protocol Label Switching)

• A virtual-circuit like technology widely used by ISPs
  • ISP sets up circuits inside their backbone ahead of time
  • ISP adds MPLS label to IP packet at ingress, undo at egress
Datagrams vs Virtual Circuits

• Complementary strengths

<table>
<thead>
<tr>
<th>Issue</th>
<th>Datagrams</th>
<th>Virtual Circuits</th>
</tr>
</thead>
<tbody>
<tr>
<td>Setup phase</td>
<td>Not needed</td>
<td>Required</td>
</tr>
<tr>
<td>Router state</td>
<td>Per destination</td>
<td>Per connection</td>
</tr>
<tr>
<td>Addresses</td>
<td>Packet carries full address</td>
<td>Packet carries short label</td>
</tr>
<tr>
<td>Routing</td>
<td>Per packet</td>
<td>Per circuit</td>
</tr>
<tr>
<td>Failures</td>
<td>Easier to mask</td>
<td>Difficult to mask</td>
</tr>
<tr>
<td>Quality of service</td>
<td>Difficult to add</td>
<td>Easier to add</td>
</tr>
</tbody>
</table>
Internetworking (IP)
• How do we connect different networks together?
  • This is called **internetworking**
  • We’ll look at how IP does it
How Networks May Differ

• Basically, in a lot of ways:
  • Service model (datagrams, VCs)
  • Addressing (what kind)
  • QOS (priorities, no priorities)
  • Packet sizes
  • Security (whether encrypted)

• Internetworking hides the differences with a common protocol.
Connecting Datagram and VC networks

- An example to show that it’s not so easy
  - Need to map destination address to a VC and vice-versa
  - A bit of a “road bump”, e.g., might have to set up a VC

Bump! Bump!
Internet Reference Model

• Internet Protocol (IP) is the “narrow waist”
  • Supports many different links below and apps above

1. Link
   - Ethernet
   - Cable
   - DSL
   - 802.11

2. Internet
   - IP

3. Transport
   - TCP
   - UDP

4. Application
   - SMTP
   - HTTP
   - RTP
   - DNS
IP as a “Lowest Common Denominator”

• Suppose only some physical networks support QOS or security etc.
  • Difficult for internetwork to support

• Pushes IP to be a “lowest common denominator”
  • Asks little of lower-layer networks
  • Gives little as a higher layer service
### IPv4 (Internet Protocol)

- Various fields to meet straightforward needs
  - Version, Header (IHL), Total length, Protocol, and Header Checksum

![IPv4 Protocol Fields](chart)

<table>
<thead>
<tr>
<th>Field</th>
<th>Function Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Version</td>
<td>Identifies the IP version</td>
</tr>
<tr>
<td>IHL</td>
<td>Identifies the length of the header in 32-bit words</td>
</tr>
<tr>
<td>Differentiated Services</td>
<td>Identifies the quality of service</td>
</tr>
<tr>
<td>Total length</td>
<td>Identifies the length of the data payload</td>
</tr>
<tr>
<td>Identification</td>
<td>Identifies the fragment identification</td>
</tr>
<tr>
<td>D I F</td>
<td>Identifies the fragment options</td>
</tr>
<tr>
<td>Time to live</td>
<td>Identifies the time to live for the packet</td>
</tr>
<tr>
<td>Protocol</td>
<td>Identifies the protocol type</td>
</tr>
<tr>
<td>Header checksum</td>
<td>Verifies the integrity of the header</td>
</tr>
<tr>
<td>Source address</td>
<td>Identifies the source address</td>
</tr>
<tr>
<td>Destination address</td>
<td>Identifies the destination address</td>
</tr>
<tr>
<td>Options (0 or more words)</td>
<td>Identifies additional options</td>
</tr>
<tr>
<td>Payload (e.g., TCP segment)</td>
<td>Identifies the data payload</td>
</tr>
</tbody>
</table>
IPv4

- Some fields to handle packet size differences (later)
  - Identification, Fragment offset, Fragment control bits
IPv4

• Other fields to meet other needs (later, later)
  • Differentiated Services, Time to live (TTL)

Later, with QoS
Later, with ICMP

Payload (e.g., TCP segment)
**IPv4**

- Network layer of the Internet, uses datagrams
  - Provides a layer of addressing above link addresses (next)

![IPv4 Diagram](image)

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Version</td>
<td>Identifies the version of IPv4</td>
</tr>
<tr>
<td>IHL</td>
<td>Indicates the length of the header</td>
</tr>
<tr>
<td>Differentiated Services</td>
<td></td>
</tr>
<tr>
<td>Total length</td>
<td>Specifies the length of the packet</td>
</tr>
<tr>
<td>Identification</td>
<td>Unique identifier for the packet</td>
</tr>
<tr>
<td>D</td>
<td>M</td>
</tr>
<tr>
<td>Time to live</td>
<td>Indicates the time remaining</td>
</tr>
<tr>
<td>Protocol</td>
<td>Identifies the protocol used</td>
</tr>
<tr>
<td>Header checksum</td>
<td>Verifies the integrity of the packet</td>
</tr>
<tr>
<td>Source address</td>
<td>Provides the address of the sender</td>
</tr>
<tr>
<td>Destination address</td>
<td>Provides the address of the receiver</td>
</tr>
<tr>
<td>Options</td>
<td>Additional headers or options</td>
</tr>
<tr>
<td>Payload</td>
<td>The data sent with the packet</td>
</tr>
</tbody>
</table>
IP Addresses

• IPv4 uses 32-bit addresses
  • Later we’ll see IPv6, which uses 128-bit addresses

• Written in “dotted quad” notation
  • Four 8-bit numbers separated by dots

\[
\begin{align*}
\text{aaaaaa} & \text{bbbb} & \text{cccc} & \text{dddd} \\
\text{aaaaaa} & \text{bbbb} & \text{cccc} & \text{dddd} \\
00010010 & 00011111 & 00000000 & 00000001
\end{align*}
\leftrightarrow \quad \text{A.B.C.D}
\]
IP Prefixes

• Addresses are allocated in blocks called **prefixes**
  • Addresses in an L-bit prefix have the same top L bits
  • There are $2^{32-L}$ addresses aligned on $2^{32-L}$ boundary
IP Prefixes

• Written in “IP address/length” notation
  • Address is lowest address in the prefix, length is prefix bits
  • E.g., 128.13.0.0/16 is 128.13.0.0 to 128.13.255.255
  • So a /24 (“slash 24”) is 256 addresses, and a /32 is one address

00010010 00111110 00000000 xxxxxxxx ↔

↔ 128.13.0.0/16
Classful IP Addressing

• Originally, IP addresses came in fixed size blocks with the class/size encoded in the high-order bits

![Diagram showing network and host portions]

0 8 16 24 32 bits

0
10
110

Network portion Host portion

Class A, $2^{24}$ addresses
Class B, $2^{16}$ addresses
Class C, $2^{8}$ addresses
Classful IP Addressing

- This is an ARPANet assignment.
IP Forwarding

• Addresses on one network belong to a unique prefix
• Node uses a routing table that lists the next hop for prefixes

<table>
<thead>
<tr>
<th>Prefix</th>
<th>Next Hop</th>
</tr>
</thead>
<tbody>
<tr>
<td>192.24.0.0/19</td>
<td>D</td>
</tr>
<tr>
<td>192.24.12.0/22</td>
<td>B</td>
</tr>
</tbody>
</table>
Longest Matching Prefix

• Prefixes in the table might overlap!
  • Combines hierarchy with flexibility

• Longest matching prefix forwarding rule:
  • For each packet, find the longest prefix that contains the destination address, i.e., the most specific entry
  • Forward the packet to the next hop router for that prefix
Longest Matching Prefix (2)

<table>
<thead>
<tr>
<th>Prefix</th>
<th>Next Hop</th>
</tr>
</thead>
<tbody>
<tr>
<td>192.24.0.0/19</td>
<td>D</td>
</tr>
<tr>
<td>192.24.12.0/22</td>
<td>B</td>
</tr>
</tbody>
</table>

192.24.6.0 →
192.24.14.32 →
192.24.54.0 →
IP Address Work Slide:

• Route to D = 192.00011x.x.x
• Route to B = 192.00011000.000011x.x
• 192.24.6.0 = 192.00011000.00000110.00000000
• 192.24.14.32 = 192.00011000.00001110.00010000
• 192.24.54.0 = 192.00011000.00110110.00000000
## Longest Matching Prefix (2)

<table>
<thead>
<tr>
<th>Prefix</th>
<th>Next Hop</th>
</tr>
</thead>
<tbody>
<tr>
<td>192.24.0.0/19</td>
<td>D</td>
</tr>
<tr>
<td>192.24.12.0/22</td>
<td>B</td>
</tr>
</tbody>
</table>

- 192.24.6.0  →  D
- 192.24.14.32 → B
- 192.24.54.0  →  D

More specific

IP address

More specific
Host/Router Distinction

• In the Internet:
  • Routers do the routing, know way to all destinations
  • Hosts send remote traffic (out of prefix) to nearest router

Not for my network? Send it to the gateway (router)

It’s my job to know which way to go ...
Host Forwarding Table

• Give using longest matching prefix
  • 0.0.0.0/0 is a **default route** that catches all IP addresses

<table>
<thead>
<tr>
<th>Prefix</th>
<th>Next Hop</th>
</tr>
</thead>
<tbody>
<tr>
<td>My network prefix</td>
<td>Send to that IP</td>
</tr>
<tr>
<td>0.0.0.0/0</td>
<td>Send to my router</td>
</tr>
</tbody>
</table>
Host Forwarding Table

• Give using longest matching prefix
  • 0.0.0.0/0 is a default route that catches all IP addresses

<table>
<thead>
<tr>
<th>Prefix</th>
<th>Next Hop</th>
</tr>
</thead>
<tbody>
<tr>
<td>192.168.0.0/24</td>
<td>Send to that IP on eth0</td>
</tr>
<tr>
<td>0.0.0.0/0</td>
<td>192.168.0.1 on eth0</td>
</tr>
</tbody>
</table>
Flexibility of Longest Matching Prefix

• Can provide default behavior, with less specifics
  • Send traffic going outside an organization to a border router (gateway)

• Can special case behavior, with more specifics
  • For performance, economics, security, ...
Performance of Longest Matching Prefix

• Uses hierarchy for a compact table
  • Relies on use of large prefixes

• Lookup more complex than table
  • Used to be a concern for fast routers
  • Not an issue in practice these days
Issues?

• Where does this break down?
Issues?

• Where does this break down?

Bootstrapping (DHCP)
Finding Link nodes (ARP)
Really big packets (Fragmentation)
Errors in the network (ICMP)
Running out of addresses (IPv6, NAT)