L11: Protocols and Network layer

Frans Kaashoek
6.033 Spring 2012
http://web.mit.edu/6.033

Some slides are from lectures by Nick McKeown, Ion Stoica, Dina Katabi, Hari Balakrishnan, Sam Madden, and Robert Morris

Internet: Best Effort

No Guarantees:
- Variable Delay (jitter)
- Variable rate
- Packet loss
- Duplicates
- Reordering
- Maximum length

End hosts implement everything else

Email addresses, To, Cc, etc.
Reliable, flow-controlled connection

TCP
SMTP

Protocol

- Defines the structure of a conversation
- Typical a sequence of messages, each with its own header
- Examples: DHCP, DNS, UDP, SMTP, TCP, IP, ...
- Internet protocols defined in text documents (RFCs)
Layering of protocols

- Each layer adds/strips off its own header
- Each layer may split up higher-level data
- Each layer multiplexes multiple higher layers
- Each layer is (mostly) transparent to higher layers

The Internet Stack

Protocol Stack

App
Transport
Network
Link

TCP / UDP
IP
Ethernet

Email
TCP
IP
Ethernet

The Internet “Hour glass”

<table>
<thead>
<tr>
<th>App</th>
<th>Email</th>
<th>Web</th>
<th>VoIP</th>
<th>P2P</th>
<th>RTSP</th>
</tr>
</thead>
<tbody>
<tr>
<td>Transport</td>
<td>TCP</td>
<td>UDP</td>
<td>ICMP</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Network</td>
<td>IP</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Link</td>
<td>Ether</td>
<td>Sonet</td>
<td>ATM</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Fiber</td>
<td>TP</td>
<td>CAT5</td>
<td>WiFi</td>
<td>GSM</td>
</tr>
</tbody>
</table>

"Everything over IP, and IP over everything"

Link Layer

Problem:
Deliver data from one end of the link to the other

Need to address (6.02):
- Bits → Analog → Bits
- Framing
- Errors
- Medium Access Control
Network Layer:
finds a path to the destination and forwards packets along that path

- Difference between routing and forwarding
  - Routing is finding the path
  - Forwarding is the action of sending the packet to the next-hop toward its destination

Forwarding

- Each router has a forwarding table
- Forwarding tables are created by a routing protocol

Forwarding table at R

<table>
<thead>
<tr>
<th>Dest. Addr</th>
<th>Link</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>1</td>
</tr>
<tr>
<td>B</td>
<td>2</td>
</tr>
<tr>
<td>C</td>
<td>1</td>
</tr>
<tr>
<td>E</td>
<td>3</td>
</tr>
</tbody>
</table>

Inside a router

The IP Header

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Version</td>
<td>Version number</td>
</tr>
<tr>
<td>HLen</td>
<td>Header length</td>
</tr>
<tr>
<td>TOS</td>
<td>Type of Service</td>
</tr>
<tr>
<td>TTL</td>
<td>Time to Live</td>
</tr>
<tr>
<td>Protocol</td>
<td>Protocol Type</td>
</tr>
<tr>
<td>ID</td>
<td>Identification</td>
</tr>
<tr>
<td>FRAG Offset</td>
<td>FragmentationOffset</td>
</tr>
<tr>
<td>Hop count</td>
<td>Number of hops</td>
</tr>
<tr>
<td>FLAGS</td>
<td>Flags</td>
</tr>
<tr>
<td>CHECKSUM</td>
<td>Checksum</td>
</tr>
<tr>
<td>SRC IP Address</td>
<td>Source IP Address</td>
</tr>
<tr>
<td>DST IP Address</td>
<td>Destination IP Address</td>
</tr>
<tr>
<td>OPTIONS</td>
<td>Options</td>
</tr>
<tr>
<td>(PAD)</td>
<td>Padding</td>
</tr>
</tbody>
</table>
Forwarding an IP Packet

- Lookup packet’s DST in forwarding table
  - If known, find the corresponding outgoing link
  - If unknown, drop packet
- Decrement TTL and drop packet if TTL is zero; update header Checksum
- Forward packet to outgoing port
- Transmit packet onto link

And switches today...

![Switches](image)

The Routing Problem:

- Generate forwarding tables

![Routing Diagram](image)

Goals: No loops, short paths, etc.
Path Vector Routing Protocol

- Initialization
- Each node knows the path to itself

For example, D initializes its paths:

<table>
<thead>
<tr>
<th>DST</th>
<th>Link</th>
<th>Path</th>
</tr>
</thead>
<tbody>
<tr>
<td>D</td>
<td>null</td>
<td></td>
</tr>
</tbody>
</table>

A 1 2 3
B 2
C 2 3
D 1 2 3
E 3

End layer
Path Vector

- Step 1: Advertisement
  - Each node tells its neighbors its path to each node in the graph

```
A 1 1 1
D 2 1 1
E 1 2 1
B 1
C 1
```

For example, D receives:

<table>
<thead>
<tr>
<th>From</th>
<th>To</th>
<th>Path</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>null</td>
<td></td>
</tr>
<tr>
<td>C</td>
<td>null</td>
<td></td>
</tr>
<tr>
<td>E</td>
<td>null</td>
<td></td>
</tr>
</tbody>
</table>

- Step 2: Update Route Info
  - Each node uses the advertisements to update its paths

D receives:

<table>
<thead>
<tr>
<th>From</th>
<th>To</th>
<th>Path</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>null</td>
<td></td>
</tr>
<tr>
<td>C</td>
<td>null</td>
<td></td>
</tr>
<tr>
<td>E</td>
<td>null</td>
<td></td>
</tr>
</tbody>
</table>

D updates its paths:

```
A 1 1 1
D 2 1 1
E 1 2 1
B 1
C 1
```

Note: At the end of first round, each node has learned all one-hop paths

Questions About Path Vector

- How do we avoid permanent loops?
- What happens when a node hears multiple paths to the same destination?
- What happens if the graph changes?

Periodically repeat Steps 1 & 2

In round 2, D receives:

<table>
<thead>
<tr>
<th>From</th>
<th>To</th>
<th>Path</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>null</td>
<td></td>
</tr>
<tr>
<td>C</td>
<td>null</td>
<td></td>
</tr>
<tr>
<td>E</td>
<td>null</td>
<td></td>
</tr>
</tbody>
</table>

D updates its paths:

```
A 1 1 1
D 2 1 1
E 1 2 1
B 1
C 3
```

Note: At the end of round 2, each node has learned all two-hop paths
**Questions About Path Vector**

- How do we ensure no loops?
  - When a node updates its paths, it never accepts a path that has itself
- What happens when a node hears multiple paths to the same destination?
  - It picks the better path (e.g., the shorter number of hops)
- What happens if the graph changes?
  - Algorithm deals well with new links
  - To deal with links that go down, each router should discard any path that a neighbor stops advertising

**Hierarchical Routing**

- Internet: collection of domains/networks
- Inside a domain: Route over a graph of routers
- Between domains: Route over a graph of domains
- Address consists of “Domain Id”, “Node Id”

**Hierarchical Routing**

**Advantage**
- Scalable
  - Smaller tables
  - Smaller messages
- Delegation
  - Each domain can run its own routing protocol

**Disadvantage**
- Mobility is difficult
  - Address depends on geographic location
- Sup-optimal paths
  - E.g., in the figure, the shortest path between the two machines should traverse the yellow domain.

**Routing: many open issues**

- Misconfigurations between domains?
- Flat addresses and scalable?
- Routing in multihop WiFi networks?
- Routing in peer-to-peer networks?
Summary

• Protocols
• Layering of protocols
• Network layer: forwarding & Routing
  • Path-vector routing protocol