What breaks when we keep adding people to the same wire?

Only solution: split up the people onto multiple wires
  - But how can they talk to each other?

When should these boxes forward packets between wires?
  - How do you specify a destination?
  - How does your packet find its way?
Outline

• Bridging
• Internetworks
  • Methods for packet forwarding
• Traditional IP addressing

Building Larger LANs: Bridges

• Extend reach of a single shared medium
• Connect two or more “segments” by copying data frames between them
  • Only copy data when needed → key difference from repeaters/hubs
  • Reduce collision domain compared with single LAN
  • Separate segments can send at once → much greater bandwidth
• Challenge: learning which packets to copy across links

Transparent Bridges

• Design goals:
  • Self-configuring without hardware or software changes
  • Bridge do not impact the operation of the individual LANs
• Three parts to making bridges transparent:
  1) Forwarding frames
  2) Learning addresses/host locations
  3) Spanning tree algorithm

Frame Forwarding

• A machine with MAC Address lies in the direction of number port of the bridge
• For every packet, the bridge “looks up” the entry for the packets destination MAC address and forwards the packet on that port.
  • Other packets are broadcast – why?
• Timer is used to flush old entries
Learning Bridges

- Manually filling in bridge tables?
  - Time consuming, error-prone
- Keep track of source address of packets arriving on every link, showing what segment hosts are on
  - Fill in the forwarding table based on this information

Spanning Tree Bridges

- More complex topologies can provide redundancy.
  - But can also create loops.
- What is the problem with loops?
- Solution: spanning tree

Spanning Tree Protocol Overview

Embed a tree that provides a single unique path to each destination:
1) Elect a single bridge as a root bridge
2) Each bridge calculates the distance of the shortest path to the root bridge
3) Each LAN identifies a designated bridge, the bridge closest to the root. It will forward packets to the root.
4) Each bridge determines a root port, which will be used to send packets to the root
5) Identify the ports that form the spanning tree

Spanning Tree Algorithm Steps

- Root of the spanning tree is the bridge with the lowest identifier.
  - All ports are part of tree
- Each bridge finds shortest path to the root.
  - Remembers port that is on the shortest path
  - Used to forward packets
- Select for each LAN the designated bridge that has the shortest path to the root.
  - Identifier as tie-breaker
  - Responsible for that LAN
Spanning Tree Algorithm

- Each node sends configuration message to all neighbors.
  - Identifier of the sender
  - Id of the presumed root
  - Distance to the presumed root
  - E.g. B5 sends (B5, B5, 0)

- When B receives a message, it decides whether the solution is better than their local solution.
  - A root with a lower identifier?
  - Same root but lower distance?
  - Same root, distance but sender has lower identifier?

- After convergence, each bridge knows the root, distance to root, root port, and designated bridge for each LAN.

Spanning Tree Algorithm (part 2)

- Each bridge B can now select which of its ports make up the spanning tree:
  - B’s root port
  - All ports for which B is the designated bridge on the LAN

- Bridges can not configure their ports.
  - Forwarding state or blocked state, depending on whether the port is part of the spanning tree

- Root periodically sends configuration messages and bridges forward them over LANs they are responsible for.

Spanning Tree Algorithm Example

- Node B2:
  - Sends (B2, B2, 0)
  - Receives (B1, B1, 0) from B1
  - Sends (B2, B1, 1) "up"
  - Continues the forwarding forever

- Node B1:
  - Will send notifications forever

- Node B7:
  - Sends (B7, B7, 0)
  - Receives (B1, B1, 0) from B1
  - Sends (B7, B1, 1) "up" and "right"
  - Receives (B5, B5, 0) - ignored
  - Receives (B5, B1, 1) - better
  - Continues forwarding the B1 messages forever to the "right"

Problem 2 – Bridging Weaknesses

- Doesn’t handle incompatible LAN technologies
- How well does it scale?
What is an Internetwork?

- Multiple incompatible LANs can be physically connected by specialized computers called routers.
- The connected networks are called an internetwork.
  - The "Internet" is one (very big & successful) example of an internetwork.

Logical Structure of Internet

- Ad hoc interconnection of networks
  - No particular topology
  - Vastly different router & link capacities
  - Send packets from source to destination by hopping through networks
    - Router connect one network to another
    - Different paths to destination may exist

Internet Protocol (IP)

- Hour Glass Model
  - Create abstraction layer that hides underlying technology from network application software
  - Make as minimal as possible
  - Allows range of current & future technologies
  - Can support many different types of applications
Problem 3: Internetwork Design

LAN 1 ➔ router ➔ WAN ➔ router ➔ LAN 2

- How do I designate a distant host?
  • Addressing / naming
- How do I send information to a distant host?
  • What gets sent?
  • What route should it take?
- Must support:
  • Heterogeneity LAN technologies
  • Scalability ➔ ensure ability to grow to worldwide scale

Getting to a Destination

- How do you get driving directions?
- Intersections ➔ routers
- Roads ➔ links/networks
- Roads change slowly

Forwarding Packets

- Table of virtual circuits
  • Connection routed through network to set up state
  • Packets forwarded using connection state
- Source routing
  • Packet carries path
- Table of global addresses (IP)
  • Routers keep next hop for destination
  • Packets carry destination address

Simplified Virtual Circuits Example

Packet

Sender

R1

R2

R3

Receiver

Sender

R1

R2

R3

Receiver

5

2

5

2

5

1

4

3

1

4

3

1

4

3

conn 5 ➔ 4

conn 5 ➔ 3

conn 5 ➔ 3

9-21-06 Lecture 8: Bridging/Addressing/Forwarding 21

9-21-06 Lecture 8: Bridging/Addressing/Forwarding 22

9-21-06 Lecture 8: Bridging/Addressing/Forwarding 23

9-21-06 Lecture 8: Bridging/Addressing/Forwarding 25
Virtual Circuits

- Advantages
  - Efficient lookup (simple table lookup)
  - Can reserve bandwidth at connection setup
  - Easier for hardware implementations
- Disadvantages
  - Still need to route connection setup request
  - More complex failure recovery – must recreate connection state
- Typical use → fast router implementations
  - ATM – combined with fix sized cells
  - MPLS – tag switching for IP networks

Source Routing

- Advantages
  - Switches can be very simple and fast
- Disadvantages
  - Variable (unbounded) header size
  - Sources must know or discover topology (e.g., failures)
- Typical uses
  - Ad-hoc networks (DSR)
  - Machine room networks (Myrinet)

Global Address Example

Packet

Sender → R1 → R2 → R3 → Receiver

9-21-06 Lecture 8: Bridging/Addressing/Forwarding
Global Addresses

- Advantages
  - Stateless – simple error recovery
- Disadvantages
  - Every switch knows about every destination
    - Potentially large tables
  - All packets to destination take same route
  - Need routing protocol to fill table

Comparison

<table>
<thead>
<tr>
<th></th>
<th>Source Routing</th>
<th>Global Addresses</th>
<th>Virtual Circuits</th>
</tr>
</thead>
<tbody>
<tr>
<td>Header Size</td>
<td>Worst</td>
<td>OK – Large address</td>
<td>Best</td>
</tr>
<tr>
<td>Router Table Size</td>
<td>None</td>
<td>Number of hosts (prefixes)</td>
<td>Number of circuits</td>
</tr>
<tr>
<td>Forward Overhead</td>
<td>Best</td>
<td>Prefix matching (Worst)</td>
<td>Pretty Good</td>
</tr>
<tr>
<td>Setup Overhead</td>
<td>None</td>
<td>None</td>
<td>Connection Setup</td>
</tr>
<tr>
<td>Error Recovery</td>
<td>Tell all hosts</td>
<td>Tell all routers</td>
<td>Tell all routers and Tear down circuit and re-route</td>
</tr>
</tbody>
</table>

Problem 3: Router Table Size

- Global addressing networks (e.g., Internet, Ethernet bridging) require switches/routers to know next hop for all destinations
- How do we avoid large tables?

Outline

- Bridging
- Internetworks
  - Methods for packet forwarding
- Traditional IP addressing
Addressing in IP

- IP addresses are names of interfaces
  - E.g., 128.2.1.1
- Domain Name System (DNS) names are names of hosts
  - E.g., www.cmu.edu
- DNS binds host names to interfaces
- Routing binds interface names to paths

Router Table Size

- One entry for every host on the Internet
  - 630M (1/09) entries, doubling every 2.5 years
- One entry for every LAN
  - Every host on LAN shares prefix
  - Still too many and growing quickly
- One entry for every organization
  - Every host in organization shares prefix
  - Requires careful address allocation

Addressing Considerations

- Hierarchical vs. flat
  - Pennsylvania / Pittsburgh / Oakland / CMU / Seshan vs. Srinivasan Seshan: 123-45-6789 vs. (412)268-0000
  - What information would routers need to route to Ethernet addresses?
    - Need hierarchical structure for designing scalable binding from interface name to route!
  - What type of Hierarchy?
    - How many levels?
    - Same hierarchy depth for everyone?
    - Same segment size for similar partition?

IP Addresses

- Fixed length: 32 bits
- Initial classful structure (1981) (not relevant now!!!)
- Total IP address size: 4 billion
  - Class A: 128 networks, 16M hosts
  - Class B: 16K networks, 64K hosts
  - Class C: 2M networks, 256 hosts

<table>
<thead>
<tr>
<th>High Order Bits</th>
<th>Format</th>
<th>Class</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>7 bits of net, 24 bits of host</td>
<td>A</td>
</tr>
<tr>
<td>10</td>
<td>14 bits of net, 16 bits of host</td>
<td>B</td>
</tr>
<tr>
<td>110</td>
<td>21 bits of net, 8 bits of host</td>
<td>C</td>
</tr>
</tbody>
</table>
### IP Address Classes (Some are Obsolete)

<table>
<thead>
<tr>
<th>Class</th>
<th>Network ID</th>
<th>Host ID</th>
</tr>
</thead>
<tbody>
<tr>
<td>Class A</td>
<td>0</td>
<td>8-24</td>
</tr>
<tr>
<td>Class B</td>
<td>10</td>
<td>16-24</td>
</tr>
<tr>
<td>Class C</td>
<td>110</td>
<td>16-24</td>
</tr>
<tr>
<td>Class D</td>
<td>1110</td>
<td></td>
</tr>
<tr>
<td>Class E</td>
<td>1111</td>
<td>Reserved for experiments</td>
</tr>
</tbody>
</table>

### Original IP Route Lookup
- Address would specify prefix for forwarding table
  - Simple lookup
- www.cmu.edu address 128.2.11.43
  - Class B address – class + network is 128.2
  - Lookup 128.2 in forwarding table
  - Prefix – part of address that really matters for routing
- Forwarding table contains
  - List of class+network entries
  - A few fixed prefix lengths (8/16/24)
- Large tables
  - 2 Million class C networks

### Subnet Addressing RFC917 (1984)
- Class A & B networks too big
  - Very few LANs have close to 64K hosts
  - For electrical/LAN limitations, performance or administrative reasons
- Need simple way to get multiple “networks”
  - Use bridging, multiple IP networks or split up single network address ranges (subnet)
- CMU case study in RFC
  - Chose not to adopt – concern that it would not be widely supported 😊

### Subnetting
- Add another layer to hierarchy
- Variable length subnet masks
  - Could subnet a class B into several chunks

```plaintext
11111111111111111111111111111111 00000000
```

Subnet Mask
### Subnetting Example

- Assume an organization was assigned address 150.100
- Assume < 100 hosts per subnet
- How many host bits do we need?
  - Seven
- What is the network mask?
  - 11111111 11111111 11111111 10000000
  - 255.255.255.128

### Forwarding Example

- Assume a packet arrives with address 150.100.12.176
- Step 1: AND address with class + subnet mask

### Aside: Interaction with Link Layer

- How does one find the Ethernet address of a IP host?
- ARP
  - Broadcast search for IP address
    - E.g., “who-has 128.2.184.45 tell 128.2.206.138” sent to Ethernet broadcast (all FF address)
  - Destination responds (only to requester using unicast) with appropriate 48-bit Ethernet address
    - E.g., “reply 128.2.184.45 is-at 0:d0:bc:f2:18:58” sent to 0:c0:4f:d:ed:c6

### IP Address Problem (1991)

- Address space depletion
  - In danger of running out of classes A and B
  - Why?
    - Class C too small for most domains
    - Very few class A – very careful about giving them out
    - Class B – greatest problem
- Class B sparsely populated
  - But people refuse to give it back
- Large forwarding tables
  - 2 Million possible class C groups
Important Concepts

- Hierarchical addressing critical for scalable system
  - Don’t require everyone to know everyone else
  - Reduces number of updates when something changes

Some People Have Too Much Time…

- Everything I needed to know about networks I learned from TV Google video
  - Ethernet collision animation

AND…..

- Just to make sure…
  1. Packets really can’t catch fire. That is not why we have insulation on wires
  2. Don’t answer “what happens after a collision” on the exam/HW with “the packets catch on fire!”