CS 3700
Networks and Distributed Systems

Lecture 5: Bridging
Bridging
- How do we connect LANs?

Function:
- Route packets between LANs

Key challenges:
- Plug-and-play, self configuration
- How to resolve loops
Recap

- Originally, Ethernet was a broadcast technology
Originally, Ethernet was a broadcast technology.
Recap

- Originally, Ethernet was a broadcast technology.
Recap

- Originally, Ethernet was a broadcast technology

Pros: Simplicity
  - Hardware is stupid and cheap

Cons: No scalability
  - More hosts = more collisions = pandemonium
The Case for Bridging

- Need a device that can bridge different LANs
  - Only forward packets to intended recipients
  - No broadcast!

Diagram:
- A
- B
- C
- Hub
The Case for Bridging

- Need a device that can **bridge** different LANs
  - Only forward packets to intended recipients
  - No broadcast!

[Diagram showing network traffic routed through a hub]

Send Packet
B → C
The Case for Bridging

- Need a device that can **bridge** different LANs
  - Only forward packets to intended recipients
  - No broadcast!

Send Packet
B → C
The Case for Bridging

- Need a device that can **bridge** different LANs
  - Only forward packets to intended recipients
  - No broadcast!

![Diagram showing network with three devices connected to a hub, and another device labeled 'Bridge' connecting different LANs with labeled packets B → C and C → B.](image-url)
The Case for Bridging

- Need a device that can bridge different LANs
  - Only forward packets to intended recipients
  - No broadcast!

Diagram:
- Send Packet B → C
- Send Packet B → C

Network Diagram:
- Nodes A, B, C connected through a Hub and a Bridge.
The Case for Bridging

- Need a device that can bridge different LANs
  - Only forward packets to intended recipients
  - No broadcast!

![Diagram showing the process of bridging packets](image)
Bridging limits the size of collision domains
Bridging limits the size of collision domains
Bridging the LANs

- Bridging limits the size of collision domains
Bridging the LANs

- Bridging limits the size of collision domains
  - Vastly improves scalability
  - Question: could the whole Internet be one bridging domain?
Bridging the LANs

- Bridging limits the size of collision domains
  - Vastly improves scalability
  - Question: could the whole Internet be one bridging domain?
- Tradeoff: bridges are more complex than hubs
  - Physical layer device vs. data link layer device
  - Need memory buffers, packet processing hardware, routing tables
Bridge Internals

Bridge

Inputs

Outputs

Switch Fabric
Bridge Internals

Bridge

Inputs

Outputs

Memory buffer

Switch Fabric
Bridge Internals

Bridge

Inputs

Switch Fabric

Outputs

Makes routing decisions
Bridge Internals

Bridge

Inputs

Outputs

Switch Fabric
Bridge Internals

Bridge

Inputs

Switch Fabric

Outputs

Hub
Bridge Internals

Bridge

Inputs

Outputs

Switch Fabric

Hub
Bridges have memory buffers to queue packets
- Bridges have memory buffers to queue packets
- Bridge is intelligent, only forwards packets to the correct output
- Bridges have memory buffers to queue packets
- Bridge is intelligent, only forwards packets to the correct output
- Bridges are high performance, full N x line rate is possible
Bridges

- Original form of Ethernet switch
- Connect multiple IEEE 802 LANs at layer 2

Goals
- Reduce the collision domain
- Complete transparency
  - “Plug-and-play,” self-configuring
  - No hardware or software changes on hosts/hubs
  - Should not impact existing LAN operations
Bridges

- Original form of Ethernet switch
- Connect multiple IEEE 802 LANs at layer 2

Goals
- Reduce the collision domain
- Complete transparency
- "Plug-and-play," self-configuring
- No hardware or software changes on hosts/hubs
- Should not impact existing LAN operations

1. Forwarding of frames
2. Learning of (MAC) Addresses
3. Spanning Tree Algorithm (to handle loops)
Frame Forwarding Tables

Each bridge maintains a forwarding table

<table>
<thead>
<tr>
<th>MAC Address</th>
<th>Port</th>
<th>Age</th>
</tr>
</thead>
<tbody>
<tr>
<td>00:00:00:00:00:AA</td>
<td>1</td>
<td>1 minute</td>
</tr>
<tr>
<td>00:00:00:00:00:BB</td>
<td>2</td>
<td>7 minutes</td>
</tr>
<tr>
<td>00:00:00:00:00:CC</td>
<td>3</td>
<td>2 seconds</td>
</tr>
</tbody>
</table>
Frame Forwarding Tables

- Each bridge maintains a **forwarding table**

<table>
<thead>
<tr>
<th>MAC Address</th>
<th>Port</th>
<th>Age</th>
</tr>
</thead>
<tbody>
<tr>
<td>00:00:00:00:00:AA</td>
<td>1</td>
<td>1 minute</td>
</tr>
<tr>
<td>00:00:00:00:00:BB</td>
<td>2</td>
<td>7 minutes</td>
</tr>
<tr>
<td>00:00:00:00:00:CC</td>
<td>3</td>
<td>2 seconds</td>
</tr>
</tbody>
</table>

![Frame Forwarding Tables](image)
Frame Forwarding Tables

- Each bridge maintains a **forwarding table**

<table>
<thead>
<tr>
<th>MAC Address</th>
<th>Port</th>
<th>Age</th>
</tr>
</thead>
<tbody>
<tr>
<td>00:00:00:00:00:AA</td>
<td>1</td>
<td>1 minute</td>
</tr>
<tr>
<td>00:00:00:00:00:BB</td>
<td>2</td>
<td>7 minutes</td>
</tr>
<tr>
<td>00:00:00:00:00:CC</td>
<td>3</td>
<td>2 seconds</td>
</tr>
</tbody>
</table>
Frame Forwarding Tables

- Each bridge maintains a **forwarding table**

<table>
<thead>
<tr>
<th>MAC Address</th>
<th>Port</th>
<th>Age</th>
</tr>
</thead>
<tbody>
<tr>
<td>00:00:00:00:00:AA</td>
<td>1</td>
<td>1 minute</td>
</tr>
<tr>
<td>00:00:00:00:00:BB</td>
<td>2</td>
<td>7 minutes</td>
</tr>
<tr>
<td>00:00:00:00:00:CC</td>
<td>3</td>
<td>2 seconds</td>
</tr>
<tr>
<td>00:00:00:00:00:DD</td>
<td>1</td>
<td>3 minutes</td>
</tr>
</tbody>
</table>
Frame Forwarding Tables

- Each bridge maintains a **forwarding table**

<table>
<thead>
<tr>
<th>MAC Address</th>
<th>Port</th>
<th>Age</th>
</tr>
</thead>
<tbody>
<tr>
<td>00:00:00:00:00:AA</td>
<td>1</td>
<td>1 minute</td>
</tr>
<tr>
<td>00:00:00:00:00:BB</td>
<td>2</td>
<td>7 minutes</td>
</tr>
<tr>
<td>00:00:00:00:00:CC</td>
<td>3</td>
<td>2 seconds</td>
</tr>
<tr>
<td>00:00:00:00:00:DD</td>
<td>1</td>
<td>3 minutes</td>
</tr>
</tbody>
</table>

Diagram of a network switch with 8 ports.
Frame Forwarding in Action

- Assume a frame arrives on port 1
Frame Forwarding in Action

- Assume a frame arrives on port 1
Frame Forwarding in Action

- Assume a frame arrives on port 1
- If the destination MAC address is in the forwarding table, send the frame on the correct output port
- Assume a frame arrives on port 1
- If the destination MAC address is in the forwarding table, send the frame on the correct output port
Frame Forwarding in Action

- Assume a frame arrives on port 1
- If the destination MAC address is in the forwarding table, send the frame on the correct output port
- If the destination MAC isn’t in the forwarding table, broadcast the frame on all ports except 1
Frame Forwarding in Action

- Assume a frame arrives on port 1
- If the destination MAC address is in the forwarding table, send the frame on the correct output port
- If the destination MAC isn’t in the forwarding table, broadcast the frame on all ports except 1
Learning Addresses

- Manual configuration is possible, but...
  - Time consuming
  - Error Prone
  - Not adaptable (hosts may get added or removed)
Learning Addresses

- Manual configuration is possible, but...
  - Time consuming
  - Error Prone
  - Not adaptable (hosts may get added or removed)
- Instead, learn addresses using a simple heuristic
  - Look at the source of frames that arrive on each port
Learning Addresses

- Manual configuration is possible, but…
  - Time consuming
  - Error Prone
  - Not adaptable (hosts may get added or removed)
- Instead, learn addresses using a simple heuristic
  - Look at the source of frames that arrive on each port
Learning Addresses

- Manual configuration is possible, but...
  - Time consuming
  - Error Prone
  - Not adaptable (hosts may get added or removed)
- Instead, learn addresses using a simple heuristic
  - Look at the source of frames that arrive on each port

<table>
<thead>
<tr>
<th>MAC Address</th>
<th>Port</th>
<th>Age</th>
</tr>
</thead>
<tbody>
<tr>
<td>00:00:00:00:00:AA</td>
<td>1</td>
<td>0 minutes</td>
</tr>
</tbody>
</table>
Learning Addresses

- Manual configuration is possible, but...
  - Time consuming
  - Error Prone
  - Not adaptable (hosts may get added or removed)

- Instead, learn addresses using a simple heuristic
  - Look at the source of frames that arrive on each port

<table>
<thead>
<tr>
<th>MAC Address</th>
<th>Port</th>
<th>Age</th>
</tr>
</thead>
<tbody>
<tr>
<td>00:00:00:00:00:AA</td>
<td>1</td>
<td>0 minutes</td>
</tr>
<tr>
<td>00:00:00:00:00:BB</td>
<td>2</td>
<td>0 minutes</td>
</tr>
</tbody>
</table>
Learning Addresses

- Manual configuration is possible, but...
  - Time consuming
  - Error Prone
  - Not adaptable (hosts may get added or removed)
- Instead, learn addresses using a simple heuristic
  - Look at the source of frames that arrive

Delete old entries after a timeout

<table>
<thead>
<tr>
<th>MAC Address</th>
<th>Port</th>
<th>Age</th>
</tr>
</thead>
<tbody>
<tr>
<td>00:00:00:00:00:AA</td>
<td>1</td>
<td>0 minutes</td>
</tr>
<tr>
<td>00:00:00:00:00:BB</td>
<td>2</td>
<td>0 minutes</td>
</tr>
</tbody>
</table>
Complicated Learning Example
Complicated Learning Example

- \(<\text{Src=}AA, \text{Dest=}FF>\)
Complicated Learning Example

- `<Src=AA, Dest=FF>`

Diagram:

- Hub
  - Port 1
  - Port 2
- Bridge 1
  - Port 1
  - Port 2
- Bridge 2
  - Port 1
  - Port 2
- AA
- BB
- CC
- DD
- EE
- FF
Complicated Learning Example

- `<Src=AA, Dest=FF>`

```
Port 1
Port 2
Hub
```

```
AA
BB
CC
DD
EE
FF
```
Complicated Learning Example

- `<Src=AA, Dest=FF>`

**Diagram:**
- Hub with ports:
  - Port 1-connected to Bridge 1
  - Port 2-connected to Bridge 2
- Bridge 1 and Bridge 2 with:
  - Port 1-connected to Hub
  - Port 2-connected to Hub

Hosts:
- AA
- BB
- CC
- DD
- EE
- FF
Complicated Learning Example

- `<Src=AA, Dest=FF>`
Complicated Learning Example

- `<Src=AA, Dest=FF>`
Complicated Learning Example

- `<Src=AA, Dest=FF>`
- `<Src=CC, Dest=AA>`
Complicated Learning Example

- `<Src=AA, Dest=FF>`
- `<Src=CC, Dest=AA>`
Complicated Learning Example

- `<Src=AA, Dest=FF>`
- `<Src=CC, Dest=AA>`
Complicated Learning Example

- `<Src=AA, Dest=FF>`
- `<Src=CC, Dest=AA>`
- `<Src=EE, Dest=CC>`

Bridge 1:
- AA: 1
- CC: 2

Bridge 2:
- AA: 1
- CC: 1
Complicated Learning Example

- `<Src=AA, Dest=FF>`
- `<Src=CC, Dest=AA>`
- `<Src=EE, Dest=CC>`
Complicated Learning Example

- `<Src=AA, Dest=FF>`
- `<Src=CC, Dest=AA>`
- `<Src=EE, Dest=CC>`
Complicated Learning Example

- `<Src=AA, Dest=FF>`
- `<Src=CC, Dest=AA>`
- `<Src=EE, Dest=CC>`
The Danger of Loops

- `<Src=AA, Dest=DD>`
The Danger of Loops

- `<Src=AA, Dest=DD>`
The Danger of Loops

- `<Src=AA, Dest=DD>`
The Danger of Loops

- `<Src=AA, Dest=DD>`
The Danger of Loops

- `<Src=AA, Dest=DD>`
The Danger of Loops

- `<Src=AA, Dest=DD>`
The Danger of Loops

- \(<\text{Src}=\text{AA}, \text{Dest}=\text{DD}>\)
The Danger of Loops

- `<Src=AA, Dest=DD>`
The Danger of Loops

- `<Src=AA, Dest=DD>`
- This continues to infinity
- How do we stop this?
The Danger of Loops

- `<Src=AA, Dest=DD>`
- This continues to infinity
  - How do we stop this?
- Remove loops from the topology
  - Without physically unplugging cables
- 802.1 uses an algorithm to build and maintain a spanning tree for routing
Spanning Tree Definition
Spanning Tree Definition
Spanning Tree Definition
Spanning Tree Definition
Spanning Tree Definition
Spanning Tree Definition

- A subset of edges in a graph that:
  - Span all nodes
  - Do not create any cycles
- This structure is a tree
Spanning Tree Definition

- A subset of edges in a graph that:
  - Span all nodes
  - Do not create any cycles
- This structure is a tree
Spanning Tree Poem

Algorhyme

I think that I shall never see
a graph more lovely than a tree.
A tree whose crucial property
is loop-free connectivity.
A tree that must be sure to span
so packet can reach every LAN.
First, the root must be selected.
By ID, it is elected.
Least-cost paths from root are traced.
In the tree, these paths are placed.
A mesh is made by folks like me,
then bridges find a spanning tree.

Radia Perlman
802.1 Spanning Tree Approach

1. Elect a bridge to be the root of the tree
2. Every bridge finds shortest path to the root
3. Union of these paths becomes the spanning tree
802.1 Spanning Tree Approach

1. Elect a bridge to be the root of the tree
2. Every bridge finds shortest path to the root
3. Union of these paths becomes the spanning tree

- Bridges exchange Configuration Bridge Protocol Data Units (BPDUs) to build the tree
  - Used to elect the root bridge
  - Calculate shortest paths
  - Locate the next hop closest to the root, and its port
  - Select ports to be included in the spanning trees
Definitions

- **Bridge ID (BID)** = <Random Number>
- **Root Bridge**: bridge with the lowest BID in the tree
- **Path Cost**: cost (in hops) from a transmitting bridge to the root
- Each port on a bridge has a unique **Port ID**
- **Root Port**: port that forwards to the root on each bridge
- **Designated Bridge**: the bridge on a LAN that provides the minimal cost path to the root
  - The designated bridge on each LAN is unique
Determining the Root

- Initially, all hosts assume they are the root

- Bridges broadcast BPDUs:
  - A new root (smallest known Root ID)
  - A new root port (what interface goes towards the root)
  - A new designated bridge (who is the next hop to root)
Comparing BPDUs

if R1 < R2: use BPDU1
else if R1 == R2 and Cost1 < Cost2: use BPDU1
else if R1 == R2 and Cost1 == Cost2 and B1 < B2: use BPDU1
else: use BPDU2
Comparing BPDUs

if \( R_1 < R_2 \): use BPDU1

else if \( R_1 == R_2 \) and \( \text{Cost}_1 < \text{Cost}_2 \): use BPDU1

else if \( R_1 == R_2 \) and \( \text{Cost}_1 == \text{Cost}_2 \) and \( B_1 < B_2 \): use BPDU1

else: use BPDU2
Comparing BPDUs

if $R_1 < R_2$: use BPDU1

else if $R_1 == R_2$ and $Cost_1 < Cost_2$: use BPDU1

else if $R_1 == R_2$ and $Cost_1 == Cost_2$ and $B_1 < B_2$: use BPDU1

else: use BPDU2
Comparing BPDUs

if R1 < R2: use BPDU1
else if R1 == R2 and Cost1 < Cost2: use BPDU1
else if R1 == R2 and Cost1 == Cost2 and B1 < B2: use BPDU1
else: use BPDU2
Comparing BPDUs

if $R_1 < R_2$: use BPDU1
else if $R_1 == R_2$ and $\text{Cost}_1 < \text{Cost}_2$: use BPDU1
else if $R_1 == R_2$ and $\text{Cost}_1 == \text{Cost}_2$ and $B_1 < B_2$: use BPDU1
else: use BPDU2
Spanning Tree Construction

0: 0/0
12: 12/0
3: 3/0
27: 27/0
41: 41/0
9: 9/0
68: 68/0
Spanning Tree Construction
Spanning Tree Construction
Spanning Tree Construction

0: 0/0

12: 0/1

3: 3/0

27: 0/1

41: 3/1

9: 3/2

68: 3/2
Spanning Tree Construction

0: 0/0

12: 0/1

3: 0/2

27: 0/1

41: 0/2

9: 3/2

68: 3/2
Spanning Tree Construction
Bridges vs. Switches

- Bridges make it possible to increase LAN capacity
  - Reduces the amount of broadcast packets
  - No loops
Bridges vs. Switches

- Bridges make it possible to increase LAN capacity
  - Reduces the amount of broadcast packets
  - No loops

- Switch is a special case of a bridge
  - Each port is connected to a single host
    - Either a client machine
    - Or another switch
  - Links are full duplex
  - Simplified hardware: no need for CSMA/CD!
  - Can have different speeds on each port
Switching the Internet

- Capabilities of switches:
  - Network-wide routing based on MAC addresses
  - Learn routes to new hosts automatically
  - Resolve loops
Switching the Internet

- Capabilities of switches:
  - Network-wide routing based on MAC addresses
  - Learn routes to new hosts automatically
  - Resolve loops

- Could the whole Internet be one switching domain?
Switching the Internet

- Capabilities of switches:
  - Network-wide routing based on MAC addresses
  - Learn routes to new hosts automatically
  - Resolve loops
- Could the whole Internet be one switching domain? NO
Limitations of MAC Routing

- Inefficient
  - Flooding packets to locate unknown hosts
Limitations of MAC Routing

- Inefficient
  - Flooding packets to locate unknown hosts
- Poor Performance
  - Spanning tree does not balance load
  - Hot spots
Limitations of MAC Routing

- Inefficient
  - Flooding packets to locate unknown hosts

- Poor Performance
  - Spanning tree does not balance load
  - Hot spots

- Extremely Poor Scalability
  - Every switch needs every MAC address on the Internet in its routing table!
Limitations of MAC Routing

- Inefficient
  - Flooding packets to locate unknown hosts

- Poor Performance
  - Spanning tree does not balance load
  - Hot spots

- Extremely Poor Scalability
  - Every switch needs every MAC address on the Internet in its routing table!

- IP addresses these problems (next week…)