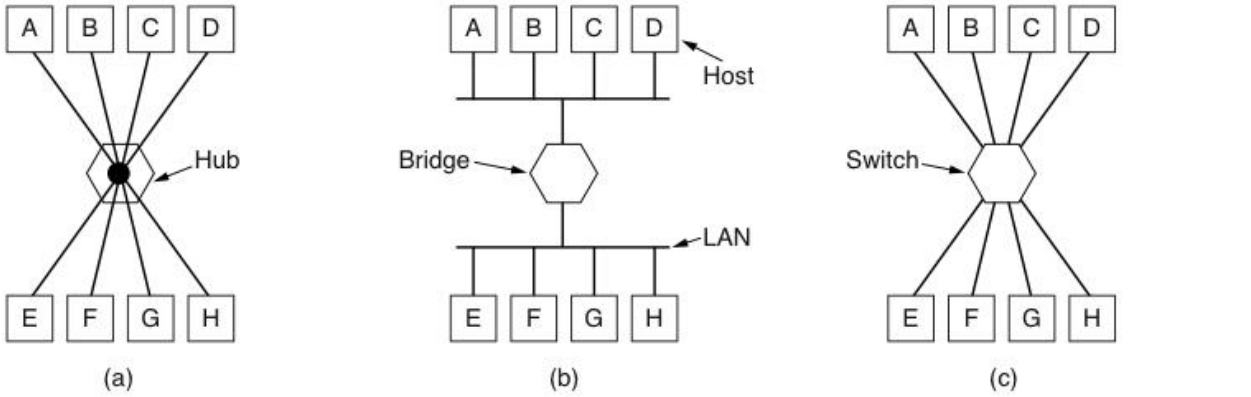
Lecture 16
Plan: Network connection devices: repeaters, hubs, bridges, and switches. Spanning tree

Review:
Physical layer: bits (how to transmit data): modulation, multiplexing
Data link layer: frames (how to divide bits into frames): framing, error control, flow control,
MAC layer: Sharing (how to share a channel): contention models, collision free models, wireless channel protocols, Ethernet and 802.11 applications

Today: talk about some of the devices used in networks and focus on spanning tree algorithm used in 802.1D

Repeaters, hubs → physical level
bridge, switch → data link level
routers → network level

Devices:



Repeater

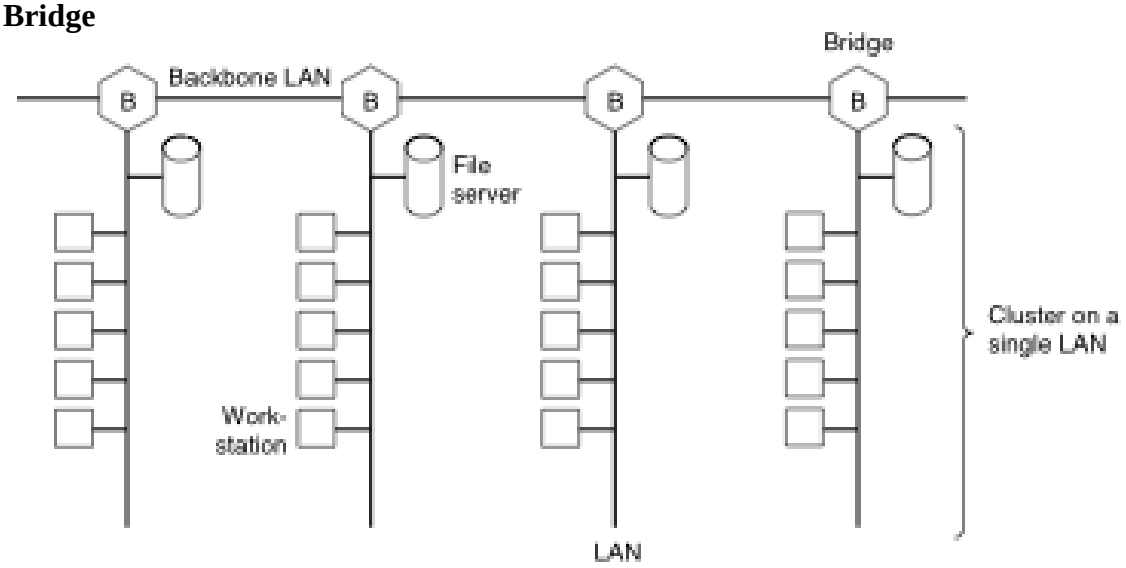
- Purpose: amplify and "clean up" the signal
- It simply sends what it hears
- The result is one LAN rather than connecting two LANs
- All hosts belong to the same collision domain

Hubs

- Purpose: to implement a broadcast network on a physical star topology
- Each host is connected to the hub with two wires (I/O). Everything the hub hears (from any input lines) will be broadcasted to all its outgoing lines

- All hosts belong to the same collision domain

Bridge



- Purpose: Connect two LANs. The LANs may use different standards
- Responsibility:
    - **a.** Translating the frame structure
        - **i.** strips one LAN's framing and adds anothers
        - **ii.** It doesn't look at the content of the frame → it only works at the data link level
        - **iii.** Problem: framesize
    - **b.** Routing
        - **i.** Forward frames received from one LAN to destinations on another LAN.
        - **ii.** Each node has a unique global MAC address, so that is not a problem.

Q:/ How to do routing?

a. Source routing bridge
    - Hosts are responsible for routing their messages through bridges
    - Each host on each LAN must know (which LAN the destination host is on, and which route to take)
    - Q:/ What if a host doesn't know
      A:/ sends a discovery frame with the destination address. Every bridge sends it to every LAN. If the destination host got the message, it sends back an ACK. on the way back, each bridge appends its (LAN#, BRIDGE#) so when the source receives it, it knows the correct route.

    Pro: Once the sender knows the path, routing is easy
    Con: more work for the host (bridge are not transparent to hosts)
    Not used in Ethernet
b. Transparent bridges
    - Bridge is transparent to the hosts (plug-and-play)
    - Each bridge keeps a table mapping host addresses to LANs.
    - Initially it is empty, it is updated each time a frame is received. If a message with source address A is received on LAN X, then (A,X) is recorded on the table

- Entries have timer → keep the old records out
- Q:/ Given this, how does routing take place?
  A:/ Read every frame on each LAN it is connected to
    - If the destination LAN is the same as the LAN the frame comes from → discard it
    - If the destination LAN is different from the LAN the frame comes from → Look up the table,
        - If found an entry→ forward to the correct LAN
        - If not found → send it to all other LANs it is connected to (flooding)

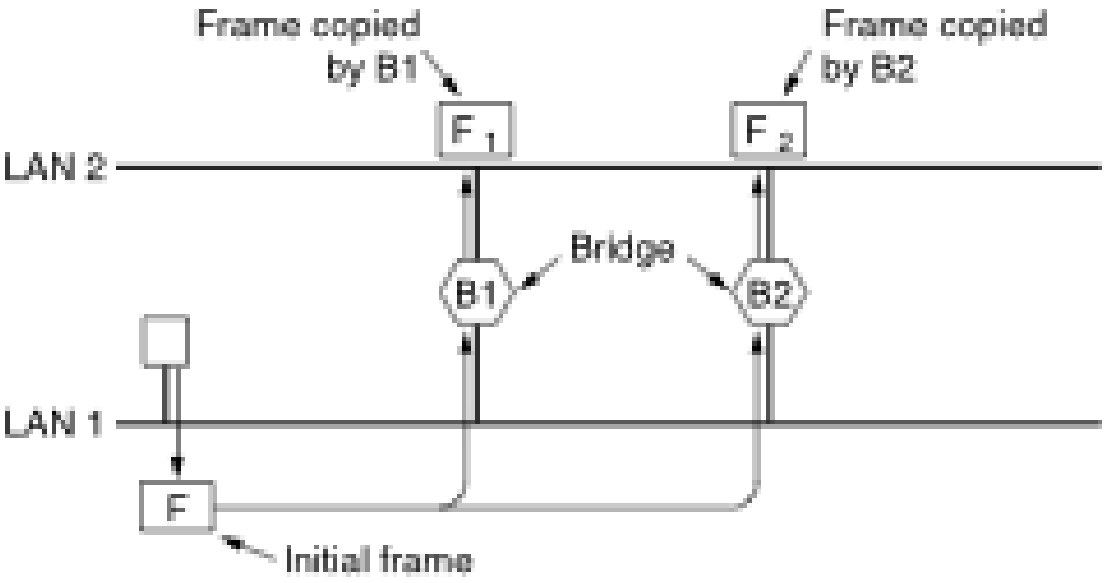
Switches

- Essentially are multi-port bridges, able to connect several LAN segments into a single LAN
- Link layer switches connects like hubs but it understand frames and addresses. Does transparent Routing
- A four-port switch has 2-way connections to 4 hosts. When it receives a frame from one host, it examines the destination address and routes it. → The frame is not broadcasted to other hosts
- Store-and-forward vs. cut-through
    - store and forward buffers an entire frame before routing
    - cut-through starts routing without receiving the entire frame
- Used in 10BaseT ethernet

Spanning tree

Q:/ Why?

A:/ With several bridges and switches in the network, there may be loops in paths

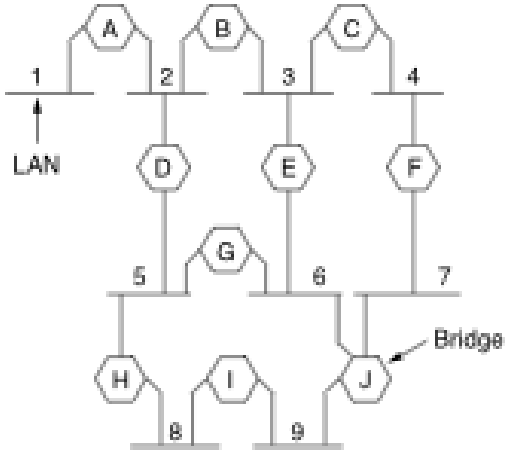


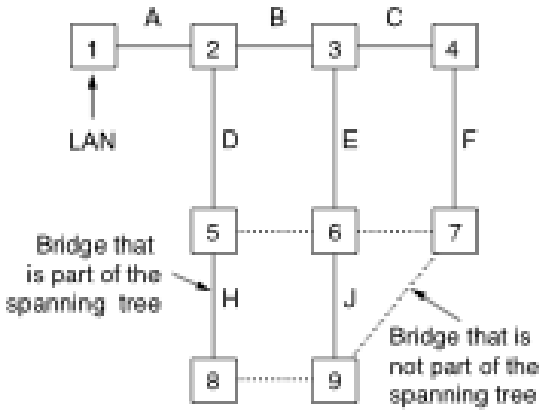
e.g. LAN1 sends frame F to an unknown host X

Then bridge B1 sends it to LAN2 as F1, so does B2 and get F2.

B1 reads B2's copy(F2) and forward it to LAN1. B2 reads B1's copy and send it to LAN2….

So IEEE standard 802.1D defines an algorithm for constructing a logical spanning tree of switches/bridges which contains no loop.





Objective:

a. The root of the spanning tree (switch with the lowest id)
b. for each bridge, which is the root port (the port which leads to the shortest path to the root)
c. For each LAN segment, which attached bridge leads to the shortest path to the root → designated switch for that segment

How:

Switches pass around messages called BPDU (bridge protocol data units). Each BPDU contains

- root id (the id of the switch with the smallest id)
- root distances (# of hops from this bridge to the destination bridge)
- source id (id of the bridge which sent this message)

BPDU are ordered lexicographically

1. In the first phase, each switch will try to indentify the root switch. Everyone builds up a BPDU with themselves as the root and sends it out to every port
If a switch receives a message with smaller root id than its own, then it knows it is not the root. → stop sending its own BPDU, but adds the message's distance by 1 and sends it to every other port. → in the end, every bridge knows which is the root id

2. After the root is determined, each switch records on the port on which the message with the least distance was received. (if there is a tie, then the source id is used to break the tie)→ This is the root port (root switch doesn't have a root port)

3. Finally, all the switches attached to a given LAN segment continue to pass around BPDUs. The one with the least distance to the root becomes the designated bridge for that segment. (source is used to break a tie). The port connecting the switch with the LAN is a designated port for that switch

All other ports on any switch other than the root port and designated ports will be blocked.

Q:/ How is it sensitive to changes in topology?
A:/

- The root bridge continues to send BPDU periodically. If a node is added or removed, the protocol will run again to reflect the new topology
- If a bridge does not receive a BPDU from the root for a while, it suspects that the root is down. It can restart the protocol by sending a BPDU with itself as a root

```
class BPDU {
   SwitchID root;
   int distance;
   SwitchID source;
}
class Switch {
   SwitchID id;
   BPDU transmit;
   List portList;
}
```

```
class Port {
   LanId lanId;
   BPDU best;
   boolean rootPort;
   boolean designatedPort;
}
```

```
if(m < p.best){
   p.best = m; // m is now the best message this port has seen
   p.designatedPort = false; // there is a better message on the LAN
}
m.distance += 1;
m.source = this.id;
if(m < transmit){
   transmit = m;
   p.root = true;
   for each port q in portList other than p {
      if(m < q.best){
         q.best = m;
         q.rootPort = false;
         q.designatedPort = true;
         write m on port q; }
   }
}
```