CPSC 471: Computer Communications

Switching and Bridging

Figures from [Computer Networks: A Systems Approach](), version 6.02dev (Larry L. Peterson and Bruce S. Davie)

Switches

- Devices that interconnect links of the same type to form a larger network
- Transfers packets from an input to one or more outputs
- Switch has fixed number of ports (I/O)
  - Limits the number of hosts that can be connected to a switch

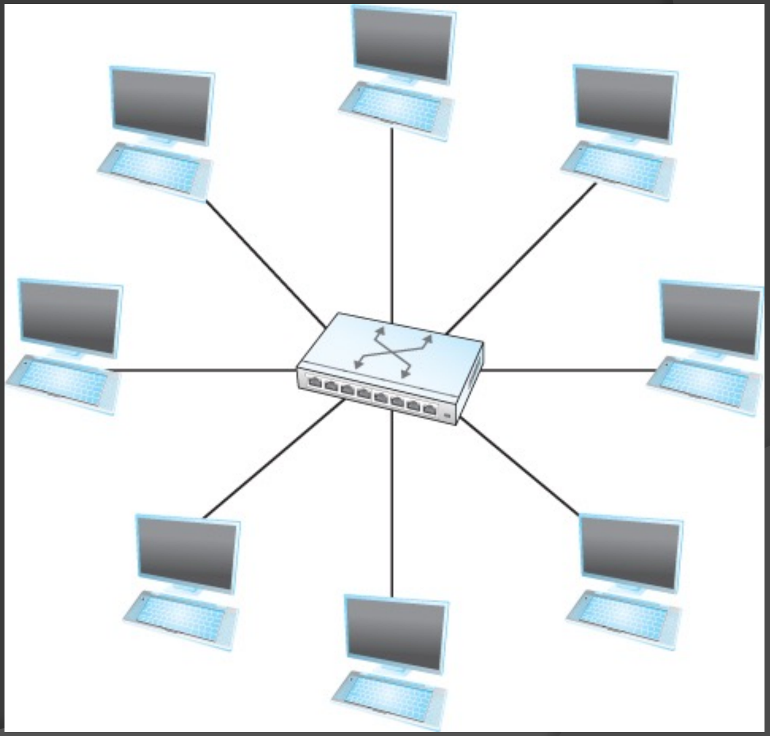


Figure 56

Packet Switching/Forwarding

- Large networks can be made by interconnecting switches
  - Switched networks are more scalable
- Switches use packet switching/forwarding
  - Receives incoming packets on one link
  - Forwards them on a different link
  - Main function of the network layer
  - How does the switch decide which output link to place a packet on?
- Three approaches to packet forwarding

Datagram/Connectionless Approach

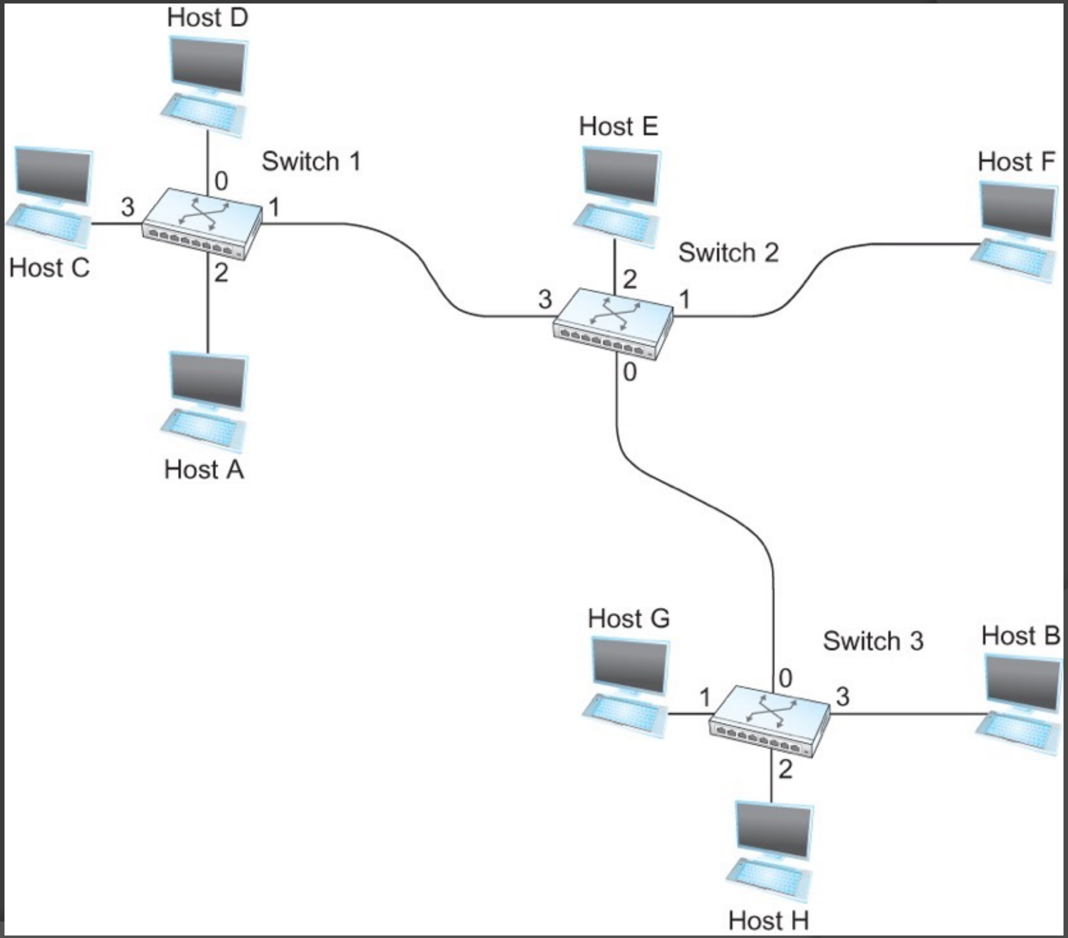
- Include enough info in every packet for any switch to decide how to get it to destination
  - Every packet contains the complete destination address
- Switch looks at a forwarding table
  - Easy to figure out for a small, simple, static network
  - Tough for a large, complex, dynamic network
    - Could contain multiple paths between a pair of hosts
    - This is routing

Forwarding Table

Figure 57

Forwarding Table for Switch 2 (Table 5)

Destination	Port
A	3
B	0
C	3
D	3
E	2
F	1
G	0
H	0



Datagram Approach continued

- A host can send a packet anywhere, anytime
  - Any packet that arrives at a switch can be immediately forwarded
- Sender does not know if network is capable to deliver it
  - Or if host is up and running
- Each packet is forwarded independently
  - Packets may use different paths between source and destination
  - This is robust to switch or link failures

Virtual Circuit (VC)/Connection-Oriented Approach

- Requires setting up a virtual connection from source to destination sending data
  - Connection setup, then data transfer
- Must establish a “connection state” in each switch between source and destination hosts
  - Consists of entry in “VC table”

VC Table

- Virtual Circuit Identifier (VCI)
  - Uniquely identifies connection at this switch
  - Carried in the header of packets of this connection
- Incoming interface that VC packets arrive at switch
- Outgoing interface that VC packets depart the switch
- A potentially different VCI for outgoing packets

VC Approach continued

- If a packet arrives on incoming interface with VCI value in header
  - Send it out on outgoing interface with outgoing VCI value placed in the header
- The VCI of received packets and the interface number uniquely identify the VC
- For a new connection, each link must assign a new VCI for the connection

Establish Connection State: PVC

- Have a network admin configure the state
  - Permanent VC or PVC
  - The VC is permanent until the network admin deletes it
- Burdensome for large networks with many switches

VC Network

VC Table entries for Switches 1-3 (Tables 6-8)

Switch #	Incoming Interface	Incoming VCI	Outgoing Interface	Outgoing VCI
1	2	5	1	11
2	3	11	2	7
3	0	7	1	4

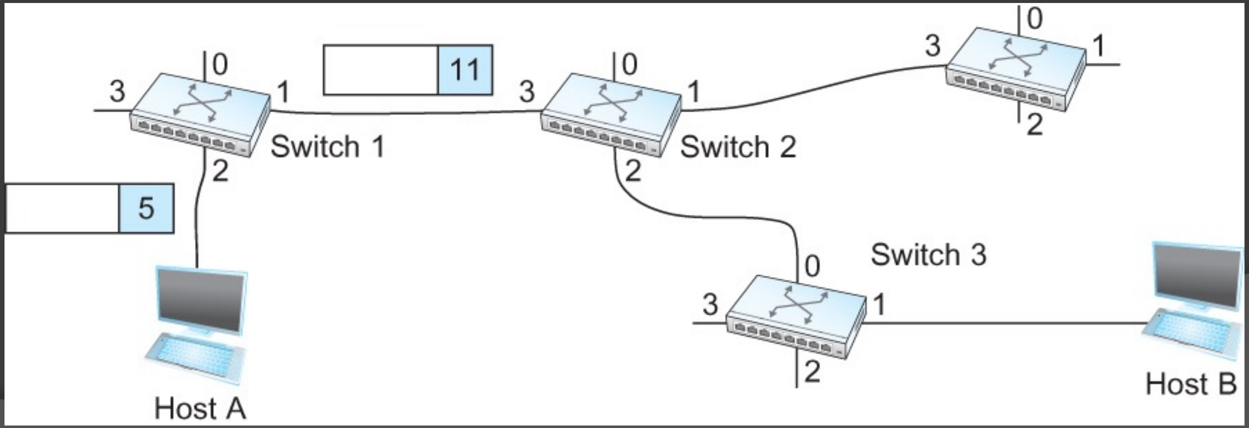


Figure 59

Establish Connection State: SVC

- A host can send messages into the network to cause the state to be established
  - Switched VC or SVC
  - This is “signaling” (resultant VC is “switched”)
  - A host may set up and delete a VC dynamically
    - Without a network admin

Signaling Example

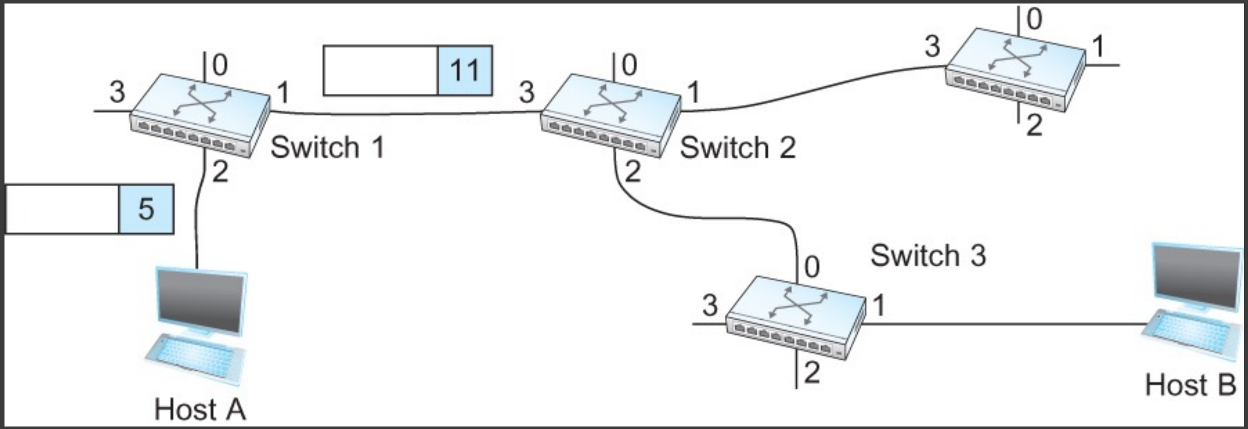


Figure 59

Summary of Signaling

- Sender sends a setup message
- Each switch receives this and sets
  - Incoming and outgoing interfaces
  - Incoming VCI (picks this value)
  - Outgoing VCI is picked by next switch down the line
- Receiver sends an ACK
- Via ACK, each switch sets the outgoing VCI
- Sender receives ACK

Teardown

- When sender no longer wishes to send data to receiver
  - Sends a teardown message
- Each switch that receives this message
  - Forwards it
  - Deletes the VC entry from its VC table

VC vs. Datagram

- Per-packet overhead of VC is less than datagram
  - Less addressing in header
- When the host receives a connection-setup ACK
  - It knows receiver is ready
  - It knows there is a route to the receiver
- If switch or link fails, connection is broken
  - A new connection must be set up
  - Old connection must be torn down
    - To clean up VC tables

VC Example: X.25

- Buffers allocated to each VC when initialized
- Sliding window protocol run between nodes
- VC is rejected if node does not have buffer space when connection request received
- These are usually called hop-by-hop flow control
  - No connection establishment for datagram
    - Each switch processes each packet independently
    - Each arriving packet competes for buffer space against other packets
- Congestion vs. Contention?

Congestion vs. Contention

- Contention
  - Packets queued at switch since they are competing for same output link
- Congestion
  - Switch has more packets than buffer space and must drop packets
- X.25 does not encounter congestion
  - Efficient use of buffer space?
- Datagram invites congestion

VC Example: ATM

- Asynchronous Transfer Mode (ATM)
  - Generic Flow Control (GFC)
  - Virtual Path Identifier (VPI)
  - Virtual Circuit Identifier (VCI)
  - Header Error Check (HEC) [8-bit CRC]
- Only one size: 53 bytes (48 bytes of data)
  - Makes switch hardware simple and efficient
  - What cell size should be picked?
- Why did ATM fail?

4	8	16	3	1	8	384 (48 bytes)
GFC	VPI	VCI	Type	CLP	HEC (CRC-8)	Payload

Figure 61

Source Routing Approach

- Info about network topology is provided by the source host
- Use an ordered list of switch ports to route packet

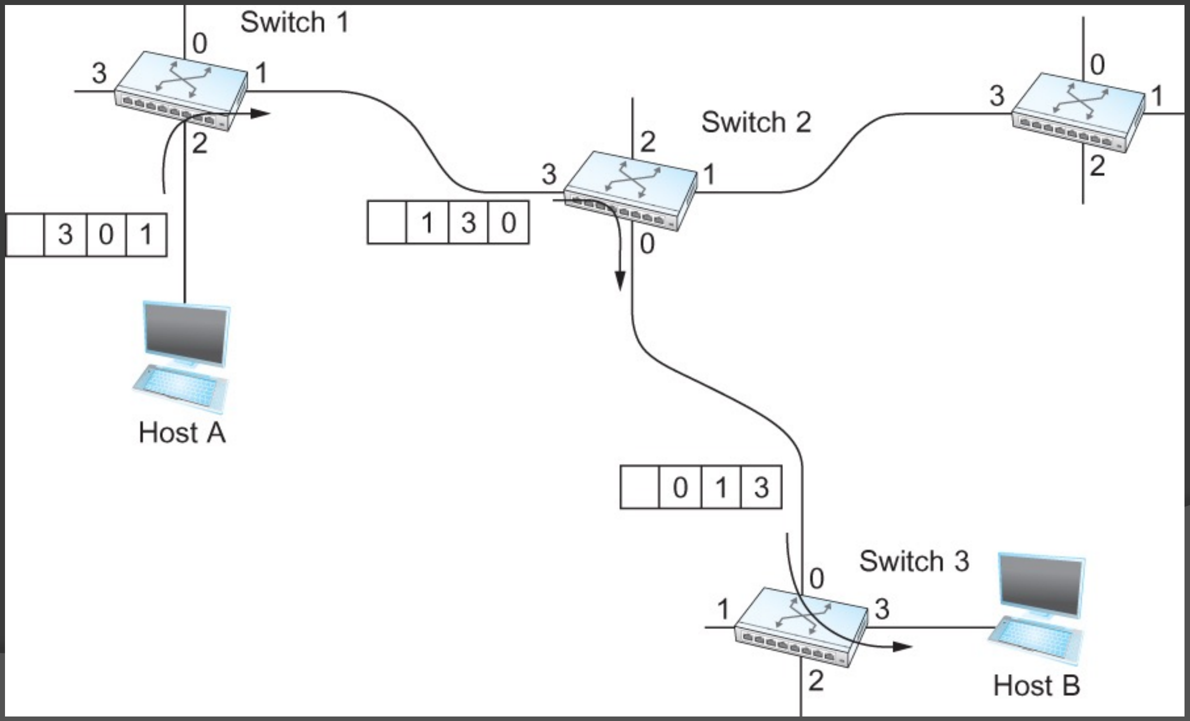
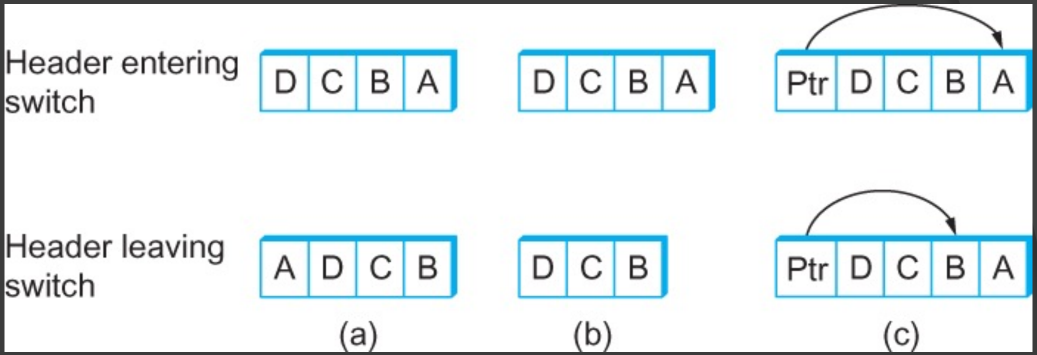


Figure 62

Source Routing continued

Figure 63



- Switch could alter header:
- Assumes the switch knows the network topology
  - Similar to building forwarding tables or figuring out where to send a setup packet
- Cannot predict how big the header will be

Strict and Loose Source Routing

- Strict
  - Every node along the path must be specified
- Loose
  - Only specifies a set of nodes to be traversed
    - Doesn’t specify exactly how to get from one node to the next
  - Can be helpful to limit the amount of info the source node must gather
    - Hard to get the complete path info in a large network

Bridges and LAN Switches

- Forward packets between LANs
- Could use a repeater between two different Ethernet segments
  - But what if this exceeds the physical limitations?
- Put a node with a pair of Ethernet adaptors between the two Ethernets
  - This is a bridge (or a LAN switch)

Ethernet example continued

- Bridge forwards frames from one Ethernet to another
  - Repeater operates on bits and blindly copies bits
- Bridge implements Ethernet’s collision detection and media access protocols on each interface
- Accepts all frames and forwards them to the other Ethernet

Bridges, more generally

- Collection of LANs connected by one or more bridges is an extended LAN
- Originally bridge accepts LAN frames on inputs and forwards them to all other outputs
- A bridge is a switch
- Provides a way to increase total bandwidth of the network

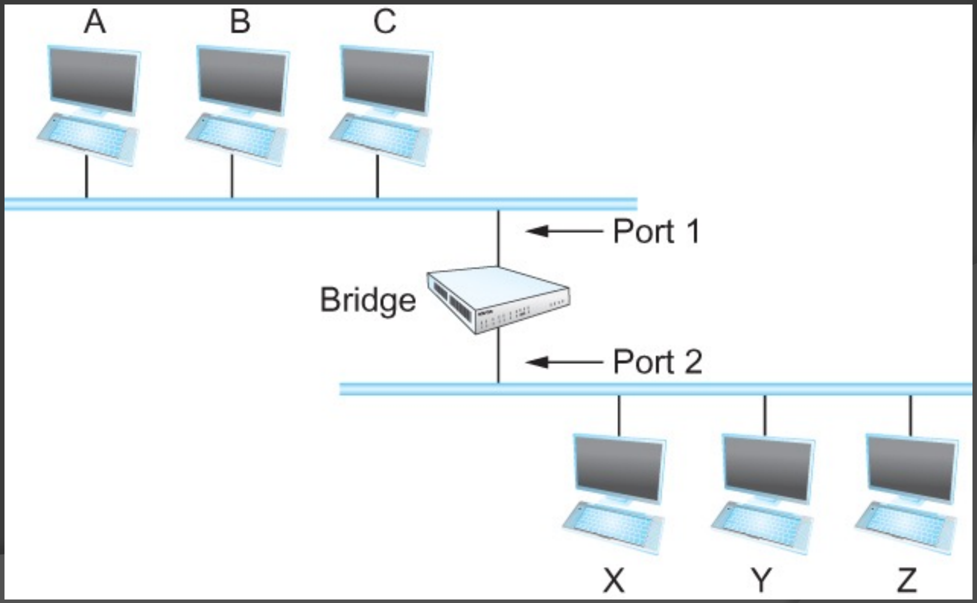
Learning Bridges

- A bridge doesn't need to forward all frames it receives
- Bridge can learn and build a forwarding table

Forwarding Table for Bridge (Table 9)

Host	Port
A	1
B	1
C	1
X	2
Y	2
Z	2

Figure 64



Learning Bridges continued

- When a bridge first boots, the table is empty
  - Entries are added over time as frames are transmitted
  - A timeout is associated with each entry, why?
  - If a bridge receives a frame for a host not in the table, it forwards the frame out on all other ports
- Table is an optimization, not required for correctness

Learning Bridge Example

- Consider the following topology:

```
                    / ----- B3 ----- C
A ----- B1 ----- B2
                    \ ____ B4 ____ D
```

- If all forwarding tables are initially empty, determine the forwarding tables for bridges B1-B4 after:
  - D sends to C
  - C sends to D
  - A sends to C

Solution to Learning Bridge Example

- All bridges see the packet from D to C
- Only B2-B4 see the packet from C to D
- Only B1-B3 see the packet from A to C

- B1: A-interface: A   B2-interface: D (not C)
- B2: B1-interface: A   B3-interface: C
  B4-interface: D
- B3: C-interface: C   B2-interface: A, D
- B4: D-interface: D   B2-interface: C (not A)

Spanning Tree Algorithm

- Learning bridges will fail if the extended LAN has a loop
  - How?
- Why would an extended LAN have a loop?
- Need a different algorithm
  - Spanning Tree

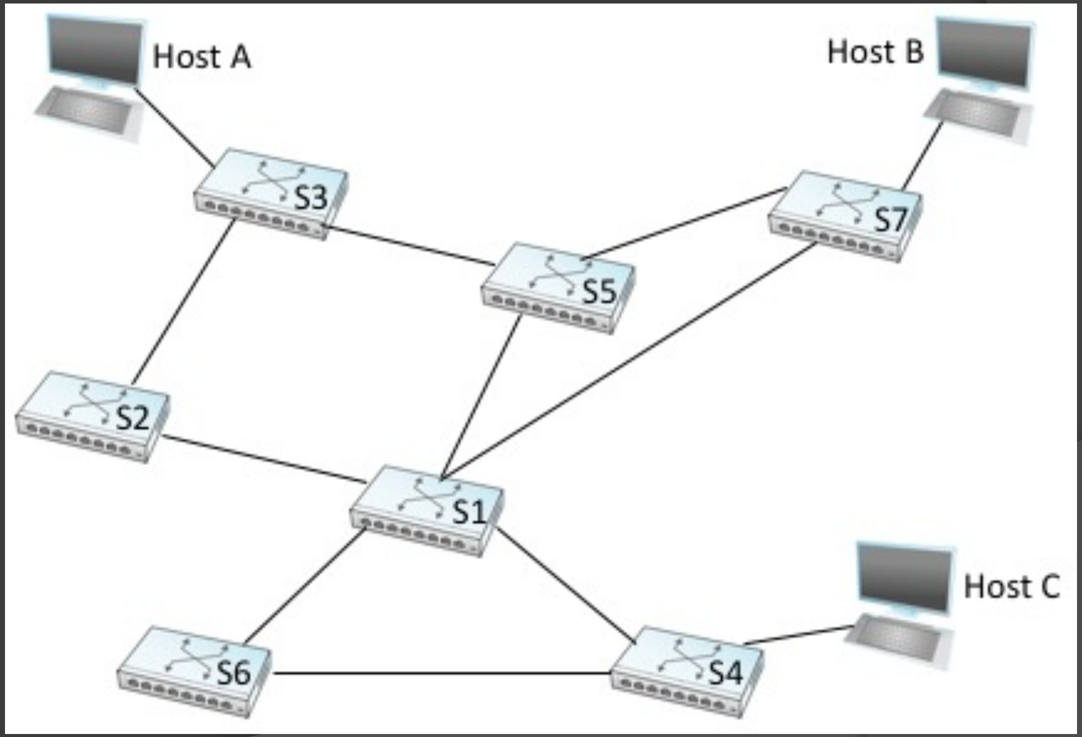


Figure 65

Spanning Tree

- A subgraph of the extended LAN's graph which covers/spans all vertices but contains no loops/cycles
- Throws out some edges of the graph

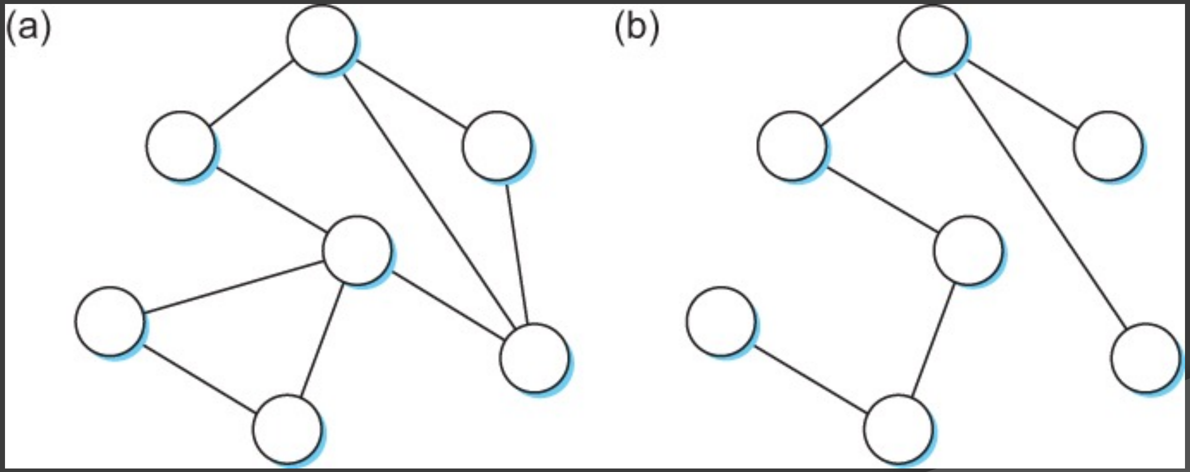


Figure 66. Cyclic Graph and Spanning Tree

Spanning Tree Algorithm

- Protocol used by bridges to agree upon a spanning tree for an extended LAN
  - Each bridge decides which ports it is and isn't willing to forward frames
  - A bridge may not participate in forwarding frames
    - Algorithm is dynamic and bridges may reconfigure themselves if a bridge fails

Spanning Tree, step-by-step

- Each bridge has a unique ID
- Bridge with smallest ID is the root bridge
  - Root bridge always forwards frames out over all its ports
- Each bridge computes shortest path to root
  - This is bridge's preferred path to root
- All bridges connected to a common LAN elect a single designated bridge that will forward frames to the root

Spanning Tree, step-by-step continued

- Electing a designated bridge
  - Designates the bridge closest to the root
    - Use smallest ID to break ties
  - Each bridge participates in election for each LAN it is connected to
- The bridge forwards frames over those ports which it is the dedicated bridge

Spanning Tree Algorithm Example

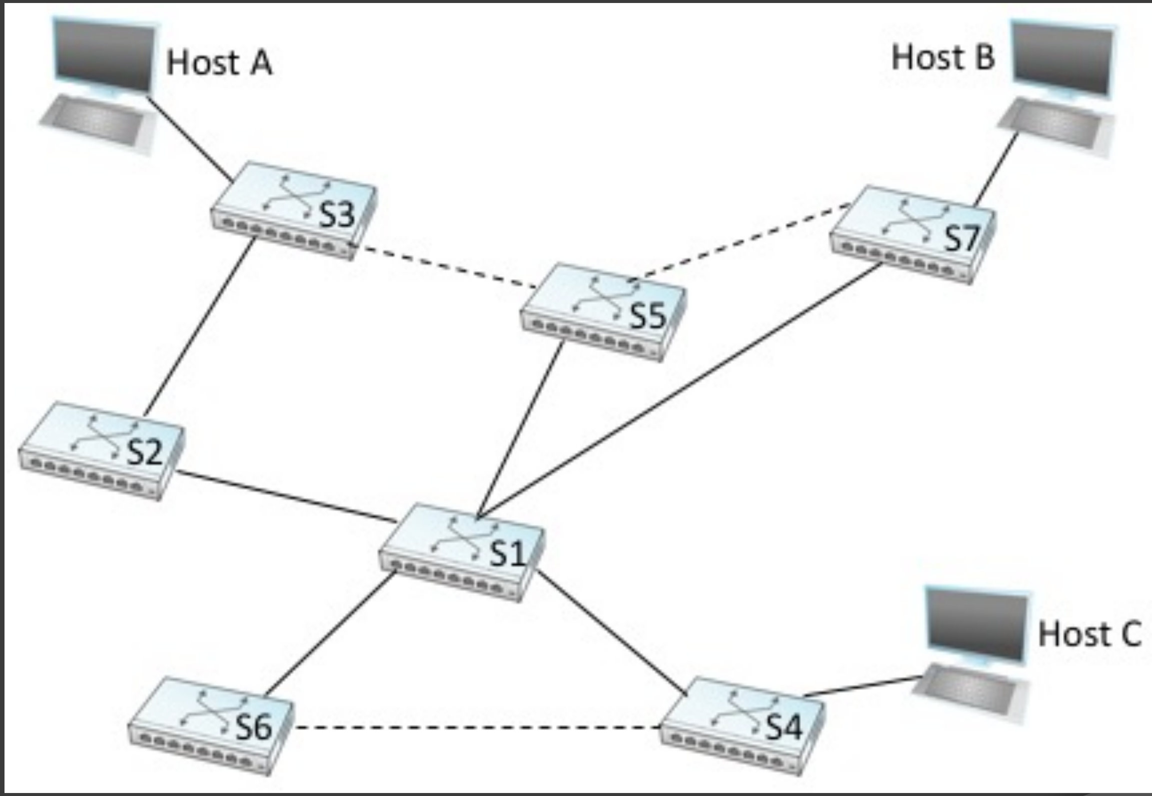


Figure 67

Spanning Tree Configuration

- Bridges must exchange configuration messages
  - Decide if they are the root or a designated bridge
- Configuration messages contain:
  - ID of bridge sending the message
  - ID of what the sending bridge believes to be the root bridge
  - The distance, in hops, from the sending bridge to the root bridge

Spanning Tree Configuration continued

- Initially each bridge thinks it's the root
  - Sends message identifying itself as root and distance to root = 0
- Each bridge records the current "best" configuration message it has sent or received
  - Better configuration message is:
    - Identifies a root with smaller ID
    - Identifies a root with equal ID, but shorter distance
    - Root ID and distance are equal, but sending bridge has a smaller ID

Spanning Tree Configuration continued

- If new configuration message better than current
  - Bridge discards current message and saves the new message
  - Adds 1 to the distance-to-root field
- Bridge stops generating configuration messages (but still forwards) if it's not the root
- Bridge stops forwarding on a port if it's not the designated bridge for that port

Spanning Tree Configuration continued

- When system stabilizes, only root bridge generates configuration messages
  - Designated bridges forward these
- Spanning tree has been built, and all bridges agree which ports will be used for the spanning tree
  - Only these ports forward packets in the extended LAN

Spanning Tree Algorithm Example

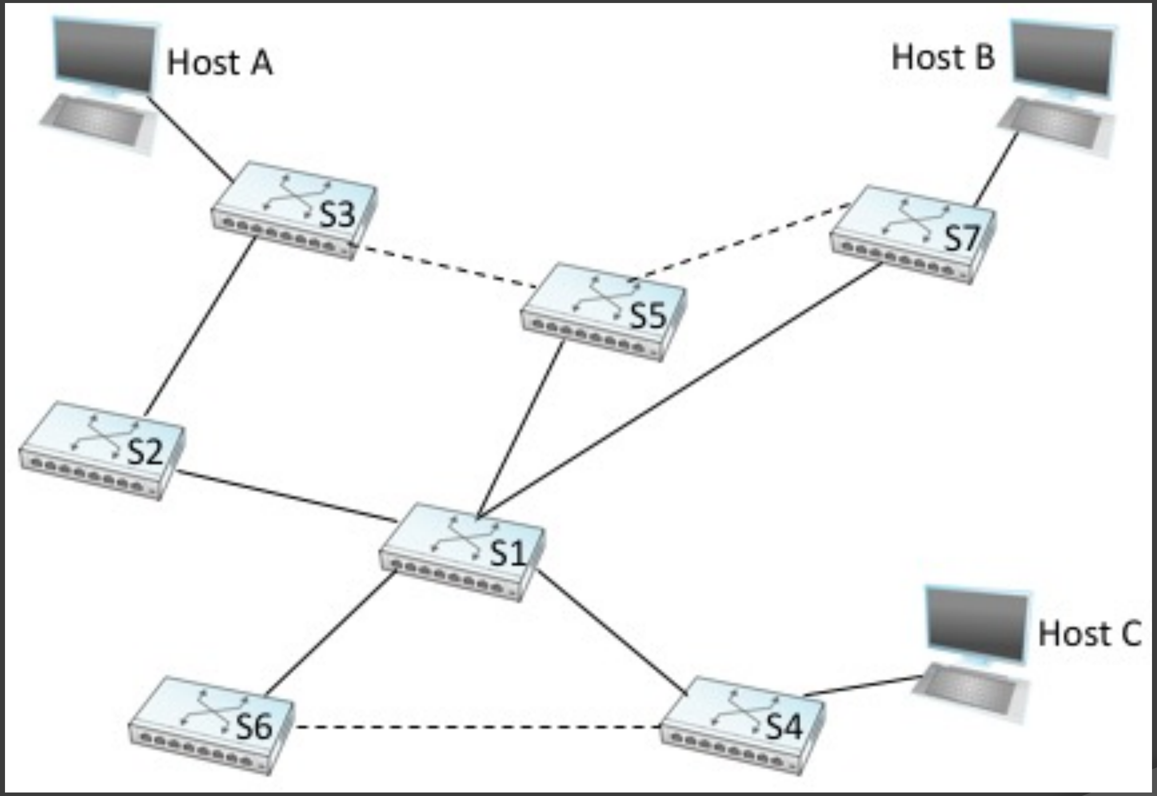


Figure 67

Spanning Tree Algorithm continued

- Root bridge continues to send configuration messages periodically
  - If a bridge fails, downstream bridges do not get these messages
    - A timeout occurs and the algorithm restarts
- Algorithm can reconfigure the spanning tree when a bridge fails
  - Cannot forward frames over alternate paths to route around a congested bridge
    - Why is this?

Broadcast and Multicast

- Bridge forwards a broadcast message to all other ports
- Multicast can do this too
  - Each host decides whether to receive the message
  - Could be done better
    - What if no multicast recipients are on the other LAN?

Limited Scalability of Bridges

- Not very scalable
  - Spanning Tree algorithm scales linearly
    - Cannot impose a hierarchy on extended LAN
  - Bridges forward all broadcast frames
    - Not every host on extended LAN may want the message
    - Broadcast does not scale
- Increase scalability by using virtual LANs

Virtual LANs (VLANs)

- Partition a single extended LAN to be into several (seemingly separate) LANs
- Each LAN is assigned an identifier
- Packets travel from one segment to another if they have the same identifier

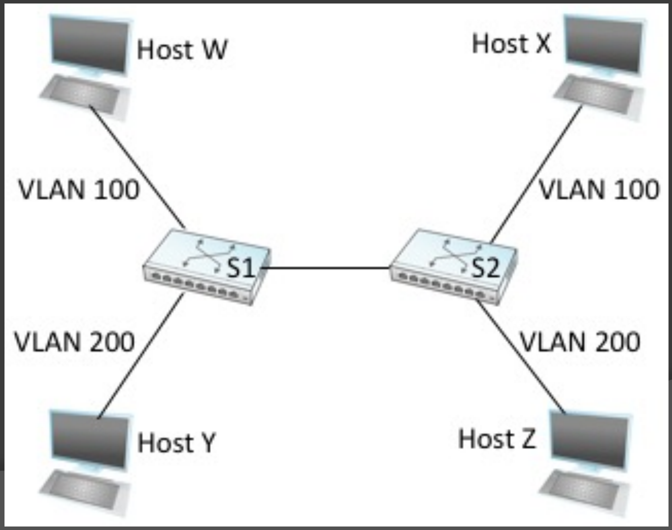


Figure 68

VLANs continued

- Limits the number of segments that receive a broadcast message
- Need to configure a VLAN ID on each port of the bridges
- Bridge observes which port a packet arrives on and which VLAN that port is configured for
- Possible to change logical topology without moving wires or changing addresses

Bridges Limited in Network Interconnection

- Meant to connect similar LANs
  - Use the network's frame header
    - Support networks with exactly the same format for addresses
      - E.g., Ethernets to Ethernets, etc.
    - Bridge between Ethernet and 802.11 is okay
      - Both use same 48-bit address format
- Do not generalize to other kinds of networks with different addressing formats
  - E.g., ATM

Bridges allow Transparent Connection of Multiple LANs

- Networks can be connected without end hosts having to run (or be aware of) any additional protocols
- Not safe to design network software assuming it will run on a single LAN
  - If a bridge becomes congestion, it'll drop frames
    - Ethernet rarely drops frames
  - Latency between a pair of hosts on an extended LAN can become larger and more variable
    - Ethernet's latency is small and predictable
  - Frames may be reordered in an extended LAN
    - Frames are never ordered on a single Ethernet