Networks and Distributed Systems

Lecture 7/8 – Switching and Bridging

Outline

- Switching and Bridging
- Basic Internetworking (IP)
- Routing

Switching and Forwarding

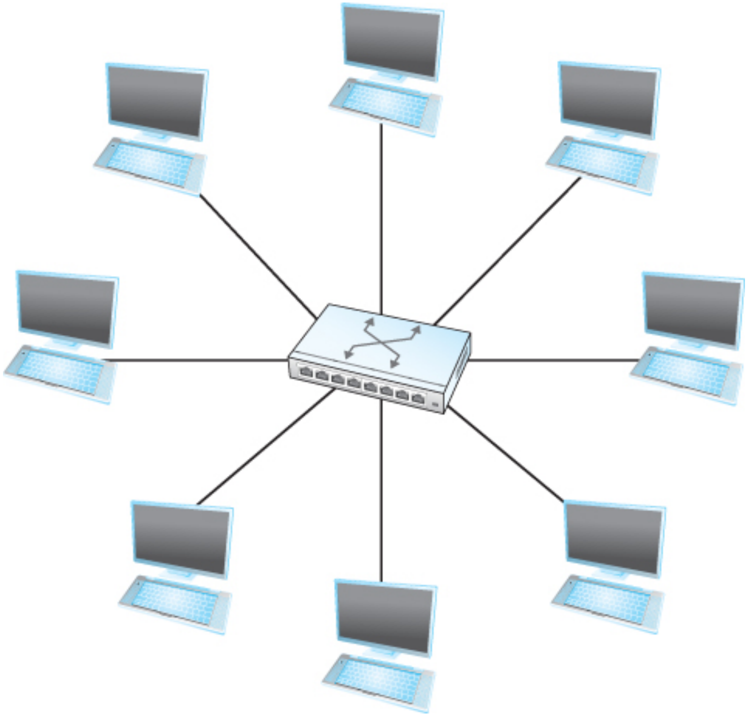
- Store-and-Forward Switches
- Bridges and Extended LANs
- Cell Switching
- Segmentation and Reassembly

Switching and Forwarding

- Switch
  - A mechanism that allows us to interconnect links to form a large network
  - A multi-input, multi-output device which transfers packets from an input to one or more outputs

Switching and Forwarding

Adds the star topology to the point-to-point link, bus (Ethernet), and ring (802.5 and FDDI) topologies



Switching and Forwarding

Properties of this star topology

- Large networks can be built by interconnecting a number of switches
  - despite that a switch has a fixed number of inputs and outputs (limits the number of hosts connected to a single switch)
- We can connect switches to each other and to hosts using point-to-point links, which typically means that we can build networks of large geographic scope
- Adding a new host to the network by connecting it to a switch does not necessarily mean that the hosts already connected will get worse performance from the network

Switching and Forwarding

The last claim cannot be made for the shared media network (discussed in previous lectures)

- It is impossible for two hosts on the same Ethernet to transmit continuously at 10Mbps because they share the same transmission medium
- Every host on a switched network has its own link to the switch
  - So it may be entirely possible for many hosts to transmit at the full link speed (bandwidth) provided that the switch is designed with enough aggregate capacity

Switching and Forwarding

- A switch is connected to a set of links and for each of these links, runs the appropriate data link protocol to communicate with that node
- A switch’s primary job is to receive incoming packets on one of its links and to transmit them on some other link
  - This function is referred as *switching and forwarding*
  - According to OSI architecture this is the main function of the network layer

Switching and Forwarding

- How does the switch decide which output port to place each packet on?
  - It looks at the header of the packet for an identifier that it uses to make the decision
  - Two common approaches
    - *Datagram or Connectionless approach*
    - *Virtual circuit or Connection-oriented approach*
  - A third approach *source routing* is less common

Switching and Forwarding

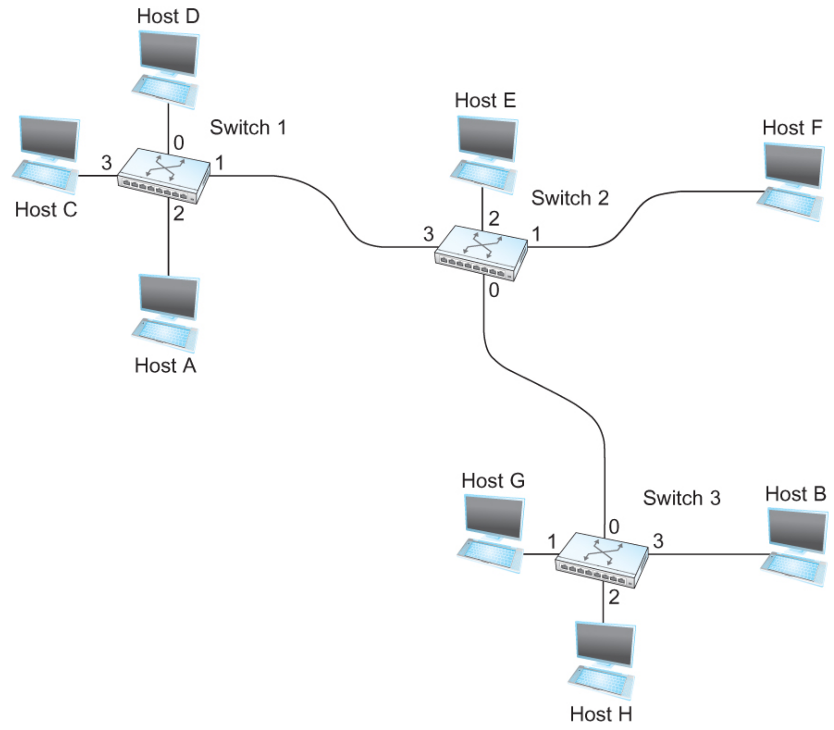
- Assumptions
  - Each host has a globally unique address
  - There is some way to identify the input and output ports of each switch
    - We can use numbers
    - We can use names

Switching and Forwarding

- Datagrams
  - Key Idea
    - Every packet contains enough information to enable any switch to decide how to get it to destination
      - Every packet contains the complete destination address

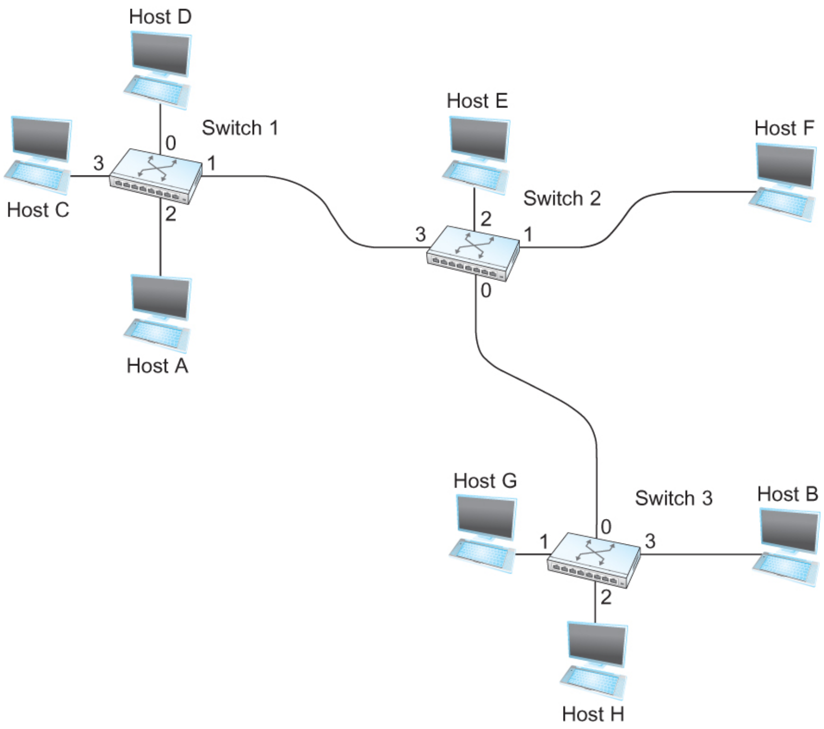
Switching and Forwarding

An example network



- To decide how to forward a packet, a switch consults a *forwarding table* (sometimes called a *routing table*)

Switching and Forwarding



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

Forwarding Table for Switch 2

Connectionless (Datagram) characteristics

- A host can send a packet anywhere at any time
- When a host sends a packet, it has no way of knowing if the network is capable of delivering it or if the destination host is even up and running
- Each packet is forwarded independently of previous packets that might have been sent to the same destination.
- A switch or link failure might not have any serious effect on communication if it is possible to find an alternate route around the failure and update the forwarding table accordingly

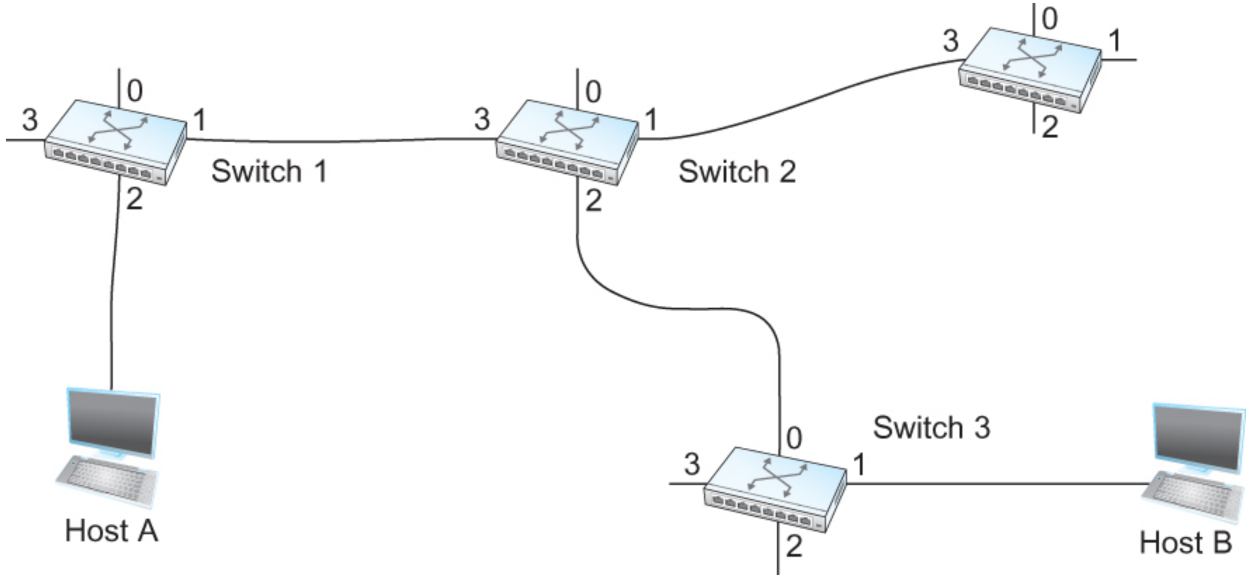
Switching and Forwarding

Virtual Circuit Switching

- Widely used technique for packet switching
- Uses the concept of *virtual circuit* (VC)
- Also called a connection-oriented model
- First set up a virtual connection from the source host to the destination host and then send the data

Switching and Forwarding

- Host A wants to send packets to host B



Switching and Forwarding

Two-stage process

- Connection setup
- Data Transfer

- Connection setup
  - Establish "connection state" in each of the switches between the source and destination hosts
  - The connection state for a single connection consists of an entry in the "VC table" in each switch through which the connection passes

Switching and Forwarding

One entry in the VC table on a single switch contains

- A virtual circuit identifier (VCI) that uniquely identifies the connection at this switch and that will be carried inside the header of the packets that belong to this connection
- An incoming interface on which packets for this VC arrive at the switch
- An outgoing interface in which packets for this VC leave the switch
- A potentially different VCI that will be used for outgoing packets

- The semantics for one such entry is
  - If a packet arrives on the designated incoming interface and that packet contains the designated VCI value in its header, then the packet should be sent out the specified outgoing interface with the specified outgoing VCI value first having been placed in its header

Note:

- The combination of the VCI of the packets as they are received at the switch and the interface on which they are received uniquely identifies the virtual connection
- There may be many virtual connections established in the switch at one time
- Incoming and outgoing VCI values are not generally the same
    - VCI is not a globally significant identifier for the connection; rather it has significance only on a given link
- Whenever a new connection is created, we need to assign a new VCI for that connection on each link that the connection will traverse
    - We also need to ensure that the chosen VCI on a given link is not currently in use on that link by some existing connection.

Switching and Forwarding

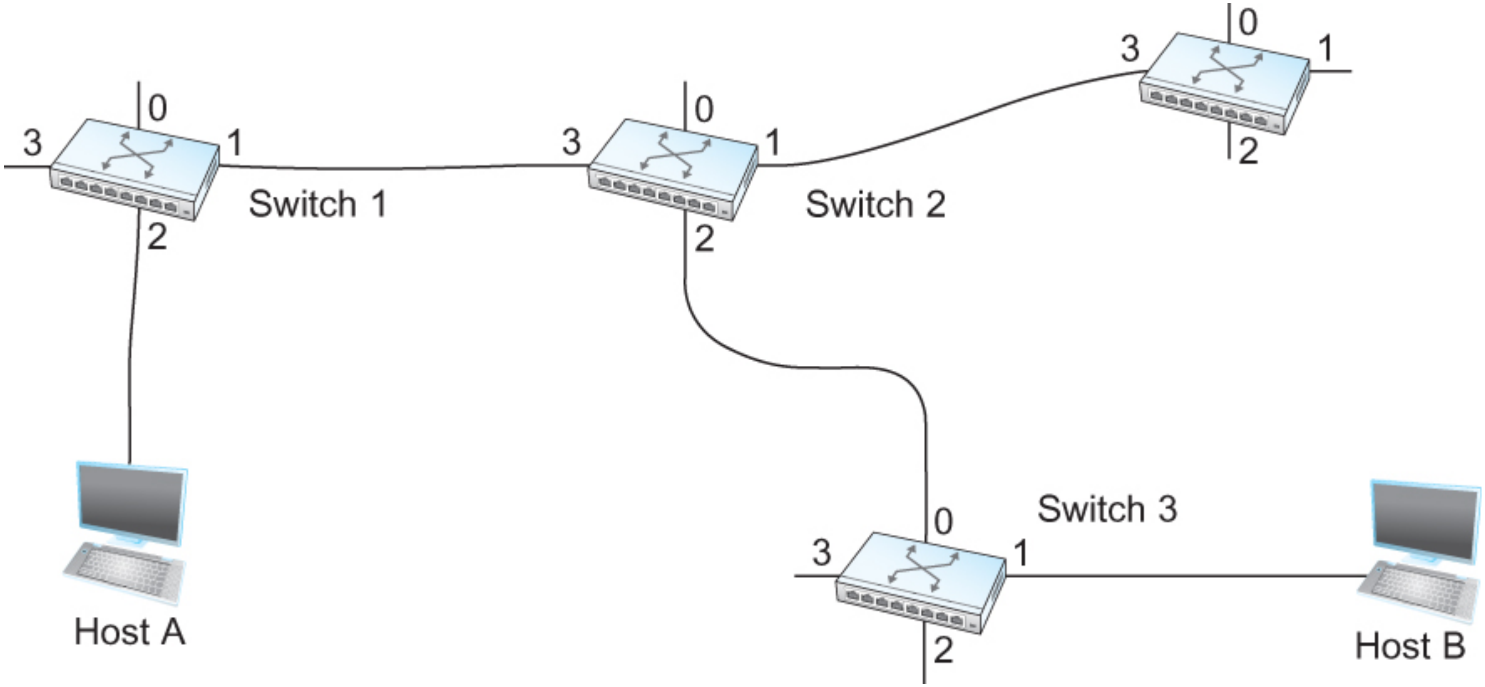
Two broad classes of approach to establishing connection state

- Network Administrator will configure the state
  - The virtual circuit is permanent (PVC)
  - The network administrator can delete this
  - Can be thought of as a long-lived or administratively configured VC
- A host can send messages into the network to cause the state to be established
  - This is referred as signalling and the resulting virtual circuit is said to be switched (SVC)
  - A host may set up and delete such a VC dynamically without the involvement of a network administrator

Switching and Forwarding

Let’s assume that a network administrator wants to manually create a new virtual connection from host A to host B

- First the administrator identifies a path through the network from A to B



Switching and Forwarding

The administrator then picks a VCI value that is currently unused on each link for the connection

- For our example,
  - Suppose the VCI value 5 is chosen for the link from host A to switch 1
  - 11 is chosen for the link from switch 1 to switch 2
  - So the switch 1 will have an entry in the VC table

Incoming Interface	Incoming VC	Outgoing Interface	Outgoing VC
2	5	1	11

Switching and Forwarding

Similarly, suppose

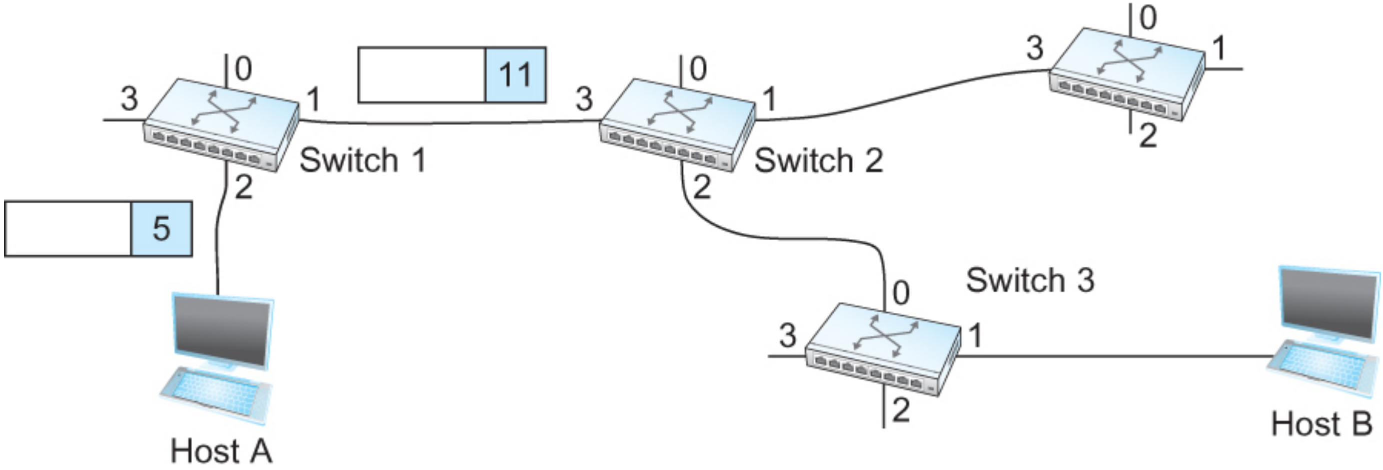
- VCI of 7 is chosen to identify this connection on the link from switch 2 to switch 3
- VCI of 4 is chosen for the link from switch 3 to host B
- Switches 2 and 3 are configured with the following VC table

Incoming Interface	Incoming VC	Outgoing Interface	Outgoing VC
3	11	2	7

Incoming Interface	Incoming VC	Outgoing Interface	Outgoing VC
0	7	1	4

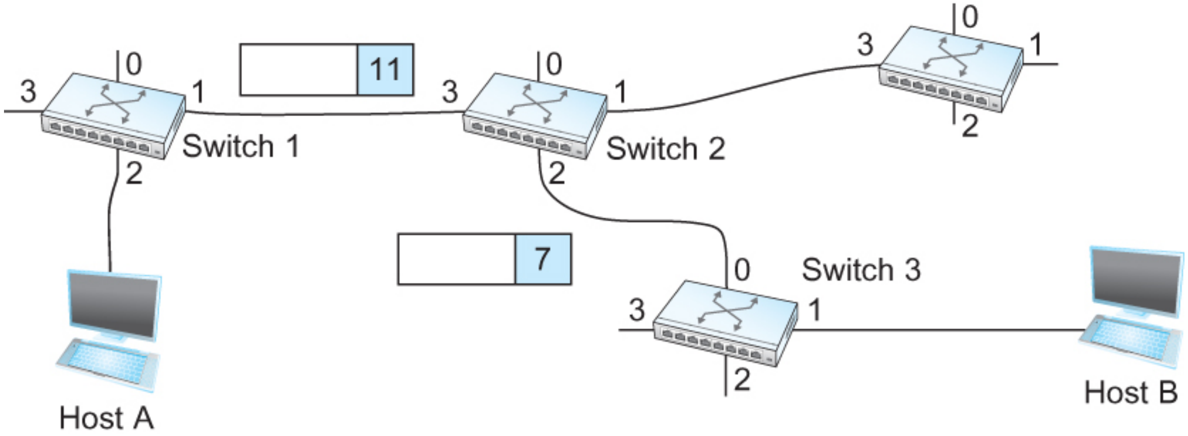
Switching and Forwarding

- For any packet that A wants to send to B, A puts the VCI value 5 in the header of the packet and sends it to switch 1
- Switch 1 receives any such packet on interface 2, and it uses the combination of the interface and the VCI in the packet header to find the appropriate VC table entry.
- The table entry on switch 1 tells the switch to forward the packet out of interface 1 and to put the VCI value 11 in the header



Switching and Forwarding

- Packet will arrive at switch 2 on interface 3 bearing VCI 11
- Switch 2 looks up interface 3 and VCI 11 in its VC table and sends the packet on to switch 3 after updating the VCI value appropriately
- This process continues until it arrives at host B with the VCI value of 4 in the packet
- To host B, this identifies the packet as having come from host A



Switching and Forwarding

- In real networks of reasonable size, the burden of configuring VC tables correctly in a large number of switches would quickly become excessive
  - Thus, some sort of signalling is almost always used, even when setting up “permanent” VCs
  - In case of PVCs, signalling is initiated by the network administrator
  - SVCs are usually set up using signalling by one of the hosts

Switching and Forwarding

- How does the signalling work
  - To start the signalling process, host A sends a setup message into the network (i.e. to switch 1)
    - The setup message contains (among other things) the complete destination address of B.
    - The setup message needs to get all the way to B to create the necessary connection state in every switch along the way
    - It is like sending a datagram to B where every switch knows which output to send the setup message so that it eventually reaches B
    - Assume that every switch knows the topology to figure out how to do that
  - When switch 1 receives the connection request, in addition to sending it on to switch 2, it creates a new entry in its VC table for this new connection
    - The entry is exactly the same shown in the previous table
    - Switch 1 picks the value 5 for this connection

Switching and Forwarding

- How does the signalling work (contd.)
  - When switch 2 receives the setup message, it performs the similar process and it picks the value 11 as the incoming VCI
  - Similarly switch 3 picks 7 as the value for its incoming VCI
    - Each switch can pick any number it likes, as long as that number is not currently in use for some other connection on that port of that switch
  - Finally the setup message arrives at host B.
  - Assuming that B is healthy and willing to accept a connection from host A, it allocates an incoming VCI value, in this case 4.
    - This VCI value can be used by B to identify all packets coming from A

Switching and Forwarding

- Now to complete the connection, everyone needs to be told what their downstream neighbor is using as the VCI for this connection
  - Host B sends an acknowledgement of the connection setup to switch 3 and includes in that message the VCI value that it chose (4)
  - Switch 3 completes the VC table entry for this connection and sends the acknowledgement on to switch 2 specifying the VCI of 7
  - Switch 2 completes the VC table entry for this connection and sends acknowledgement on to switch 1 specifying the VCI of 11
  - Finally switch 1 passes the acknowledgement on to host A telling it to use the VCI value of 5 for this connection

Switching and Forwarding

- When host A no longer wants to send data to host B, it tears down the connection by sending a teardown message to switch 1
- The switch 1 removes the relevant entry from its table and forwards the message on to the other switches in the path which similarly delete the appropriate table entries
- At this point, if host A were to send a packet with a VCI of 5 to switch 1, it would be dropped as if the connection had never existed

Characteristics of VC

- There is at least one RTT of delay before data is sent
  - Host A has to wait for the connection request to reach the far side of the network and return before it can send its first data packet
- Data packet contains only a small identifier, which is only unique on one link.
  - The per-packet overhead caused by the header is reduced relative to the datagram model
- If a switch or a link in a connection fails, the connection is broken and a new one will need to be established.
  - Also the old one needs to be torn down to free up table storage space in the switches
- The issue of how a switch decides which link to forward the connection request on has similarities with the function of a routing algorithm

Switching and Forwarding

- Good Properties of VC
  - By the time the host gets the go-ahead to send data, it knows quite a lot about the network-
    - For example, that there is really a route to the receiver and that the receiver is willing to receive data
  - It is also possible to allocate resources to the virtual circuit at the time it is established

Switching and Forwarding

- For example, an X.25 network – a packet-switched network that uses the connection-oriented model – employs the following three-part strategy
  - Buffers are allocated to each virtual circuit when the circuit is initialized
  - The sliding window protocol is run between each pair of nodes along the virtual circuit, and this protocol is augmented with the flow control to keep the sending node from overrunning the buffers allocated at the receiving node
  - The circuit is rejected by a given node if not enough buffers are available at that node when the connection request message is processed

Switching and Forwarding

- Comparison with the Datagram Model
  - Datagram network has no connection establishment phase and each switch processes each packet independently
  - Each arriving packet competes with all other packets for buffer space
  - If there are no buffers, the incoming packet must be dropped

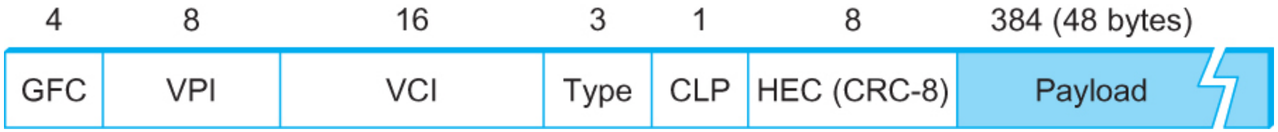
- In VC, we could imagine providing each circuit with a different quality of service (QoS)
  - The network gives the user some kind of performance related guarantee
    - Switches set aside the resources they need to meet this guarantee
      - For example, a percentage of each outgoing link's bandwidth
      - Delay tolerance on each switch
- Most popular examples of VC technologies are Frame Relay and ATM
  - One of the applications of Frame Relay is the construction of VPN

Switching and Forwarding

- ATM (Asynchronous Transfer Mode)
  - Connection-oriented packet-switched network
  - Packets are called cells
    - 5 byte header + 48 byte payload
  - Fixed length packets are easier to switch in hardware
    - Simpler to design
    - Enables parallelism

Switching and Forwarding

- ATM
  - User-Network Interface (UNI)
    - Host-to-switch format
    - GFC: Generic Flow Control
    - VCI: Virtual Circuit Identifier
    - Type: management, congestion control
    - CLP: Cell Loss Priority
    - HEC: Header Error Check (CRC-8)

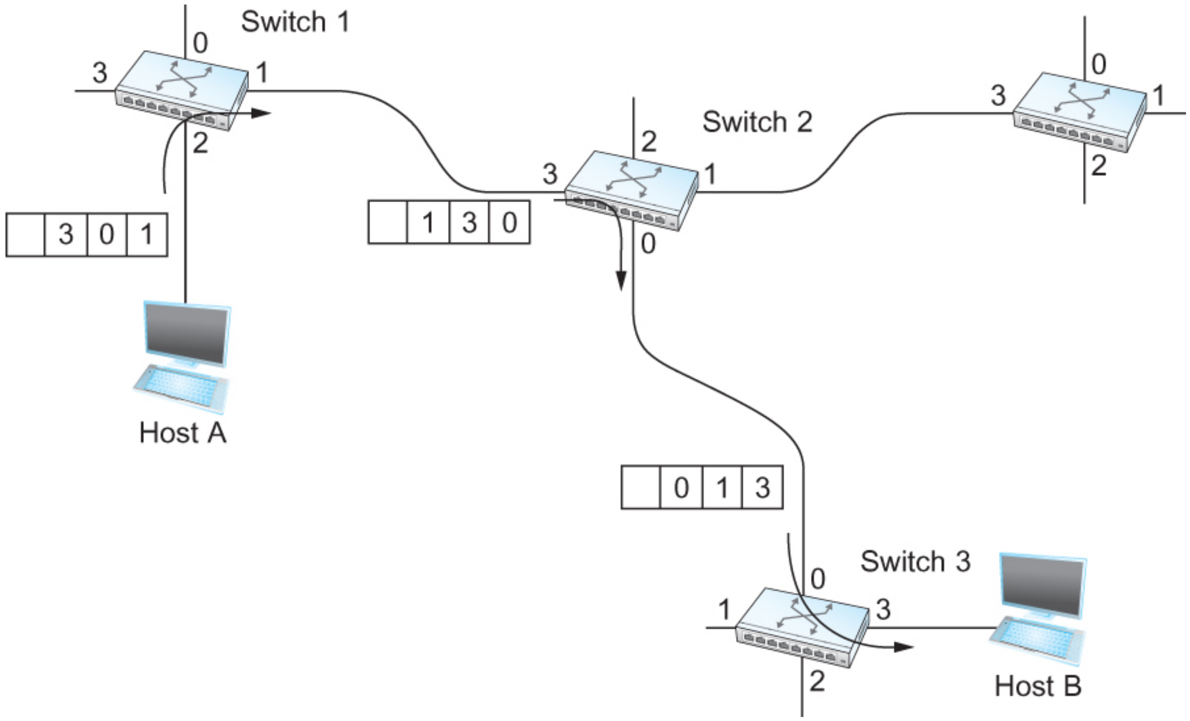


4	8	16	3	1	8	384 (48 bytes)
GFC	VPI	VCI	Type	CLP	HEC (CRC-8)	Payload

  - Network-Network Interface (NNI)
    - Switch-to-switch format
    - GFC becomes part of VPI field

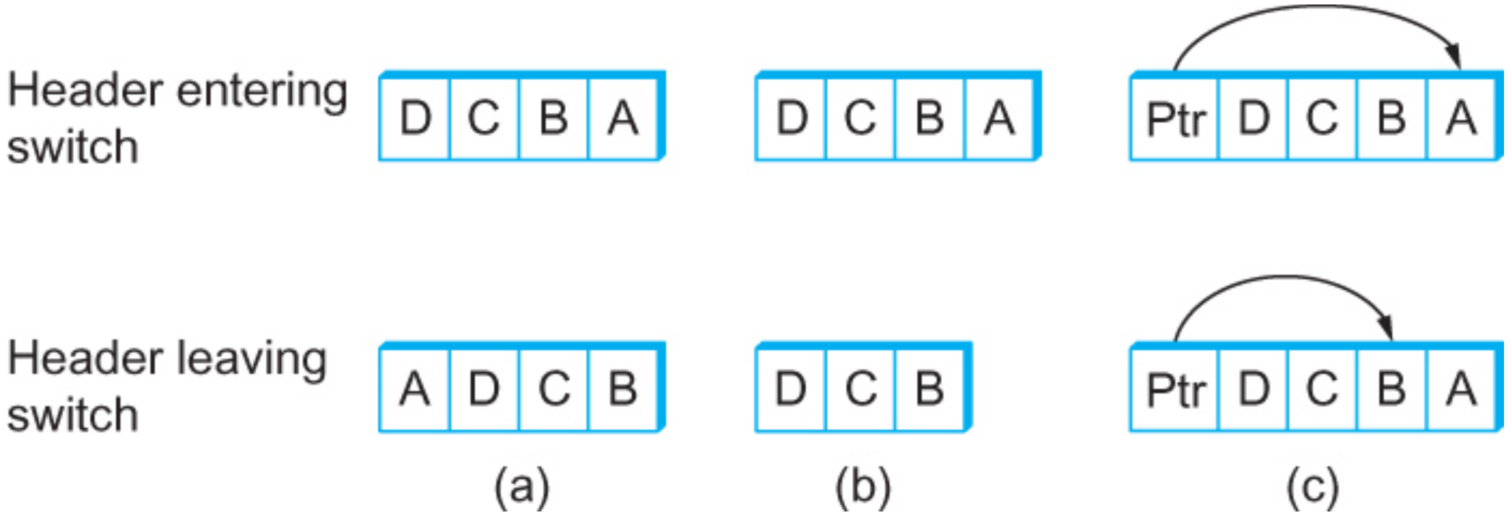
Switching and Forwarding

- Source Routing
  - All the information about network topology that is required to switch a packet across the network is provided by the source host



Switching and Forwarding

- Other approaches in Source Routing



Bridges and LAN Switches

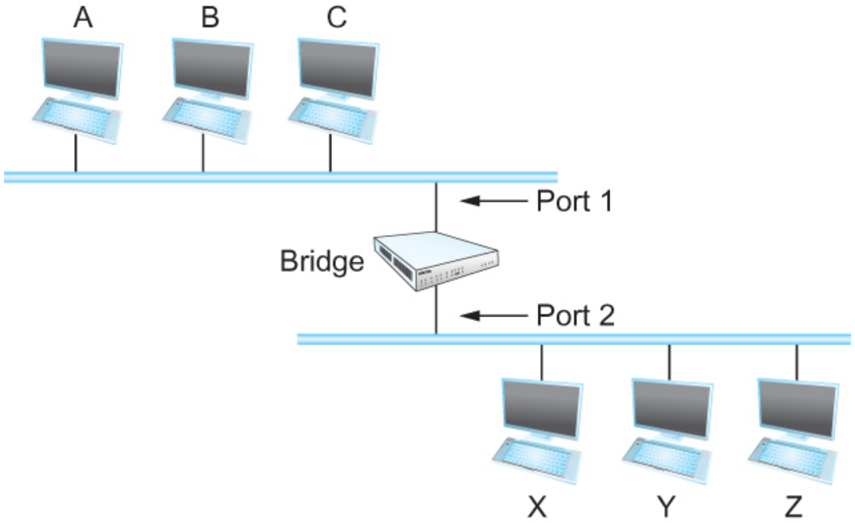
- Bridges and LAN Switches
  - Class of switches that is used to forward packets between shared-media LANs such as Ethernets
    - Known as LAN switches
    - Referred to as Bridges
  - Suppose you have a pair of Ethernets that you want to interconnect
    - One approach is put a repeater in between them
      - It might exceed the physical limitation of the Ethernet
        - No more than four repeaters between any pair of hosts
        - No more than a total of 2500 m in length is allowed
    - An alternative would be to put a node between the two Ethernets and have the node forward frames from one Ethernet to the other
      - This node is called a **Bridge**
      - A collection of LANs connected by one or more bridges is usually said to form an **Extended LAN**

Bridges and LAN Switches

- Simplest Strategy for Bridges
  - Accept LAN frames on their inputs and forward them out to all other outputs
  - Used by early bridges
- Learning Bridges
  - Observe that there is no need to forward all the frames that a bridge receives

Bridges and LAN Switches

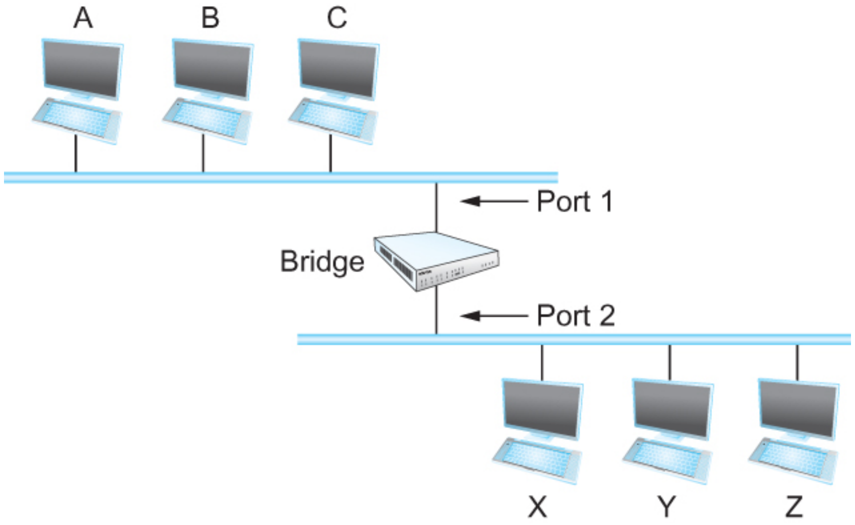
- Consider the following figure
  - When a frame from host A that is addressed to host B arrives on port 1, there is no need for the bridge to forward the frame out over port 2.



  - How does a bridge come to learn on which port the various hosts reside?

Bridges and LAN Switches

- Solution
  - Download a table into the bridge



Host	Port

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

- Who does the download?
  - Human
    - Too much work for maintenance

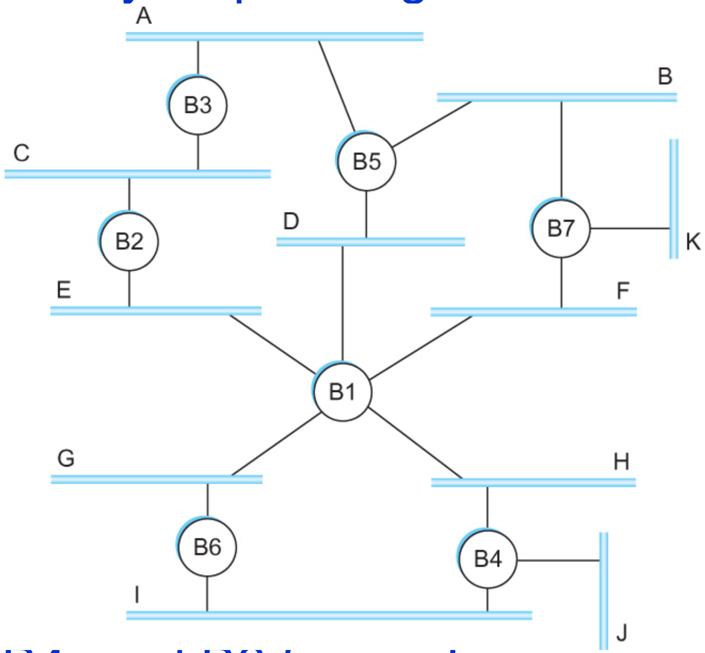
Bridges and LAN Switches

- Can the bridge learn this information by itself?
  - Yes

- How
  - Each bridge inspects the source address in all the frames it receives
  - Record the information at the bridge and build the table
  - When a bridge first boots, this table is empty
  - Entries are added over time
  - A timeout is associated with each entry
  - The bridge discards the entry after a specified period of time
    - To protect against the situation in which a host is moved from one network to another
- If the bridge receives a frame that is addressed to host not currently in the table
  - Forward the frame out on all other ports

Bridges and LAN Switches

- Strategy works fine if the extended LAN does not have a loop in it
- Why?
  - Frames potentially loop through the extended LAN forever



  - Bridges B1, B4, and B6 form a loop

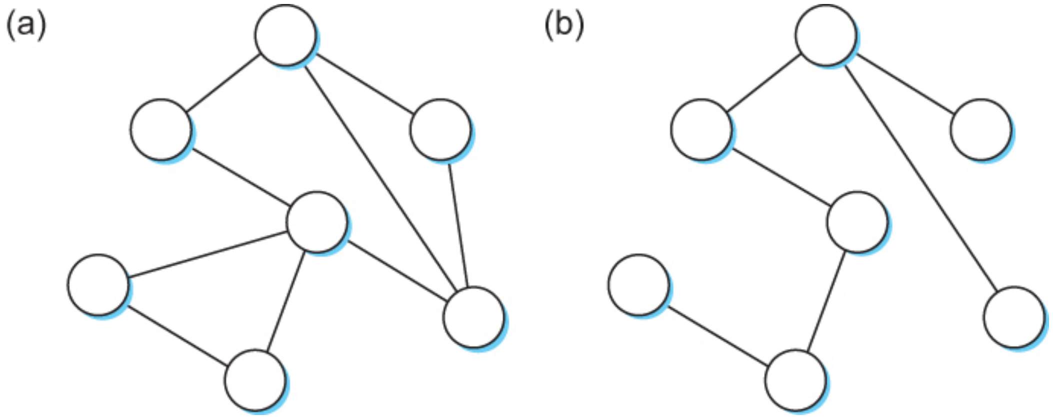
Bridges and LAN Switches

- How does an extended LAN come to have a loop in it?
  - Network is managed by more than one administrator
    - For example, it spans multiple departments in an organization
    - It is possible that no single person knows the entire configuration of the network
      - A bridge that closes a loop might be added without anyone knowing
  - Loops are built into the network to provide redundancy in case of failures
- Solution
  - Distributed Spanning Tree Algorithm

Spanning Tree Algorithm

- Think of the extended LAN as being represented by a graph that possibly has loops (cycles)

- A spanning tree is a sub-graph of this graph that covers all the vertices but contains no cycles
  - Spanning tree keeps all the vertices of the original graph but throws out some of the edges



Example of (a) a cyclic graph; (b) a corresponding spanning tree.

Spanning Tree Algorithm

- Developed by Radia Perlman at Digital
  - A protocol used by a set of bridges to agree upon a spanning tree for a particular extended LAN
  - IEEE 802.1 specification for LAN bridges is based on this algorithm
  - Each bridge decides the ports over which it is and is not willing to forward frames
    - In a sense, it is by removing ports from the topology that the extended LAN is reduced to an acyclic tree
    - It is even possible that an entire bridge will not participate in forwarding frames

Spanning Tree Algorithm

- Algorithm is dynamic
  - The bridges are always prepared to reconfigure themselves into a new spanning tree if some bridges fail

- Main idea
  - Each bridge selects the ports over which they will forward the frames

Spanning Tree Algorithm

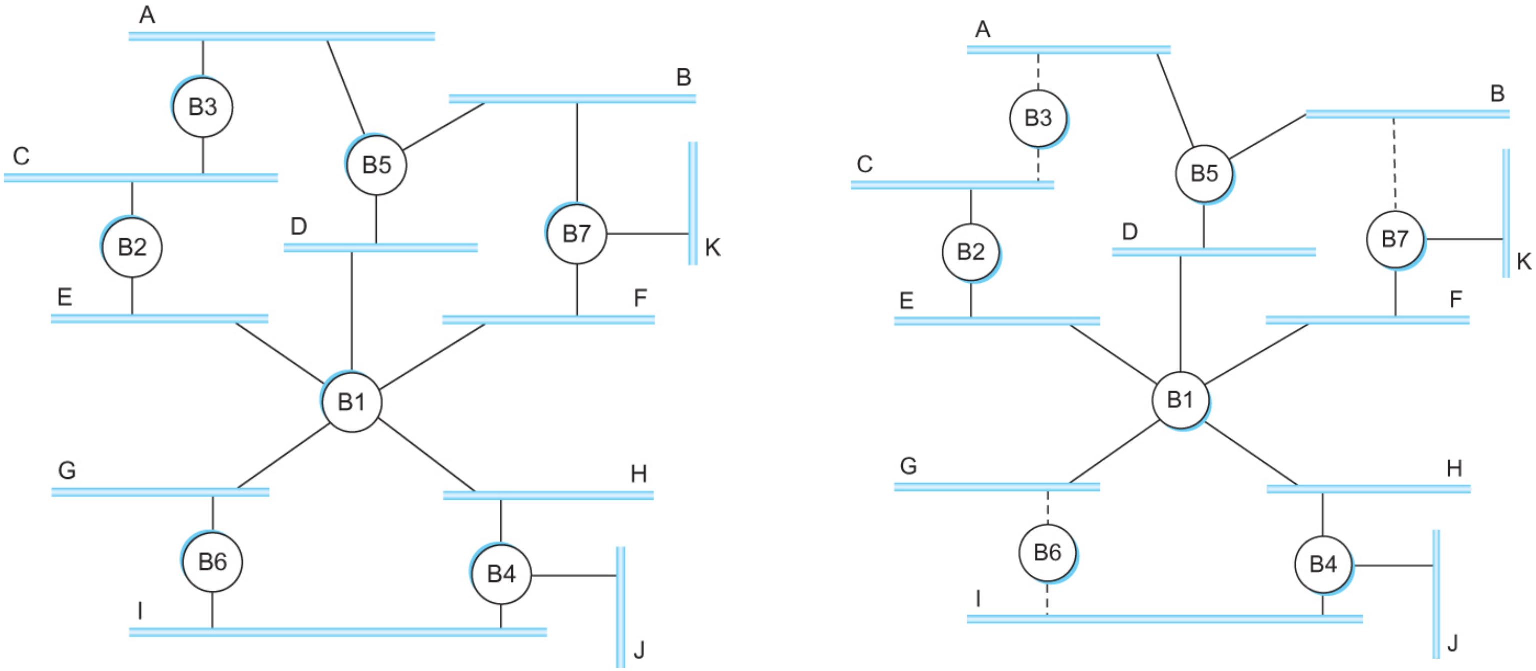
- Algorithm selects ports as follows:
  - Each bridge has a unique identifier
    - B1, B2, B3,…and so on.
  - Elect the bridge with the smallest id as the root of the spanning tree
  - The root bridge always forwards frames out over all of its ports
  - Each bridge computes the shortest path to the root and notes which of its ports is on this path
    - This port is selected as the bridge's preferred path to the root
  - Finally, all the bridges connected to a given LAN elect a single *designated bridge* that will be responsible for forwarding frames toward the root bridge

Spanning Tree Algorithm

- Each LAN's designated bridge is the one that is closest to the root
- If two or more bridges are equally close to the root,
  - Then select bridge with the smallest id
- Each bridge is connected to more than one LAN
  - So it participates in the election of a designated bridge for each LAN it is connected to.
  - Each bridge decides if it is the designated bridge relative to each of its ports
  - The bridge forwards frames over those ports for which it is the designated bridge

Spanning Tree Algorithm

- B1 is the root bridge
- B3 and B5 are connected to LAN A, but B5 is the designated bridge
- B5 and B7 are connected to LAN B, but B5 is the designated bridge



Spanning Tree Algorithm

- Initially each bridge thinks it is the root, so it sends a configuration message on each of its ports identifying itself as the root and giving a distance to the root of 0
- Upon receiving a configuration message over a particular port, the bridge checks to see if the new message is *better* than the current best configuration message recorded for that port
- The new configuration is better than the currently recorded information if
  - It identifies a root with a smaller id or
  - It identifies a root with an equal id but with a shorter distance or
  - The root id and distance are equal, but the sending bridge has a smaller id

Spanning Tree Algorithm

- If the new message is better than the currently recorded one,
  - The bridge discards the old information and saves the new information
  - It first adds 1 to the distance-to-root field

- When a bridge receives a configuration message indicating that it is not the root bridge (that is, a message from a bridge with smaller id)
  - The bridge stops generating configuration messages on its own
  - Only forwards configuration messages from other bridges after 1 adding to the distance field

Spanning Tree Algorithm

- When a bridge receives a configuration message that indicates it is not the designated bridge for that port

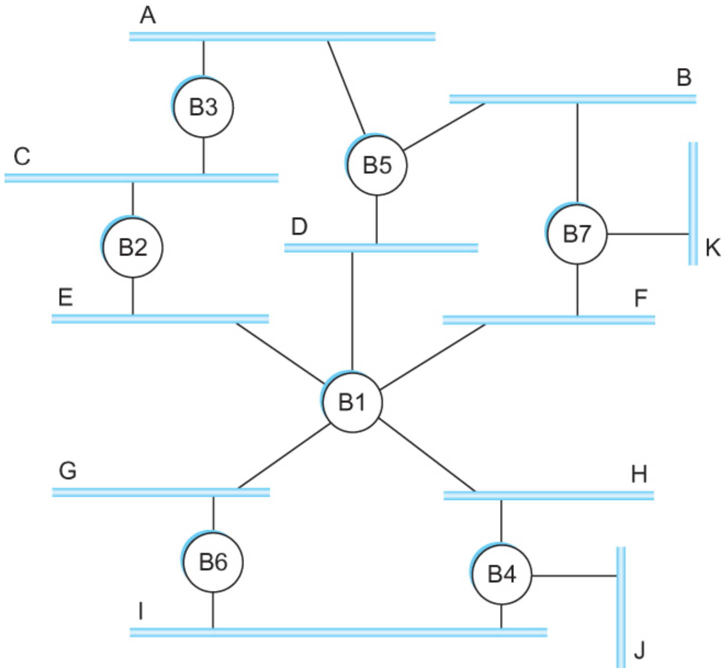
  => a message from a bridge that is closer to the root or equally far from the root but with a smaller id

    - The bridge stops sending configuration messages over that port

- When the system stabilizes,
  - Only the root bridge is still generating configuration messages.
  - Other bridges are forwarding these messages only over ports for which they are the designated bridge

Spanning Tree Algorithm

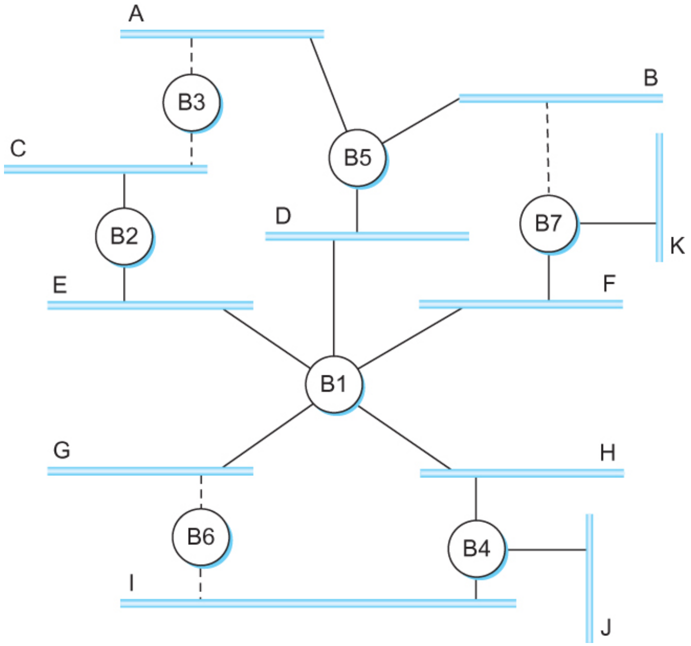
- Consider the situation when the power had just been restored to the building housing the following network



- All bridges would start off by claiming to be the root

Spanning Tree Algorithm

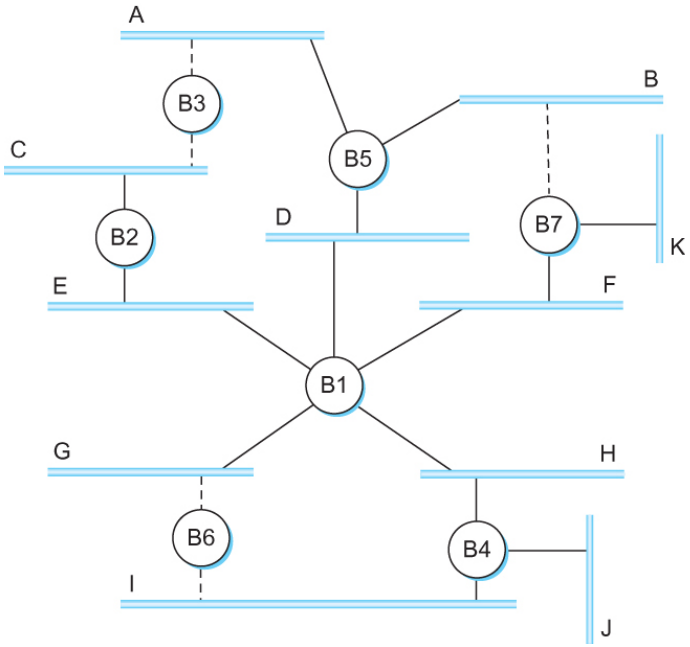
- Denote a configuration message from node X in which it claims to be distance d from the root node Y as (Y, d, X)



- Consider the activity at node B3

Spanning Tree Algorithm

- B3 receives (B2, 0, B2)
- Since 2 < 3, B3 accepts B2 as root
- B3 adds 1 to the distance advertised by B2 and sends (B2, 1, B3) to B5
- Meanwhile B2 accepts B1 as root because it has the lower id and it sends (B1, 1, B2) toward B3
- B5 accepts B1 as root and sends (B1, 1, B5) to B3
- B3 accepts B1 as root and it notes that both B2 and B5 are closer to the root than it is.
  - Thus B3 stops forwarding messages on both its interfaces
  - This leaves B3 with both ports not selected



Spanning Tree Algorithm

- Even after the system has stabilized, the root bridge continues to send configuration messages periodically
  - Other bridges continue to forward these messages
- When a bridge fails, the downstream bridges will not receive the configuration messages
- After waiting a specified period of time, they will once again claim to be the root and the algorithm starts again
- Note
  - Although the algorithm is able to reconfigure the spanning tree whenever a bridge fails, it is not able to forward frames over alternative paths for the sake of routing around a congested bridge

Spanning Tree Algorithm

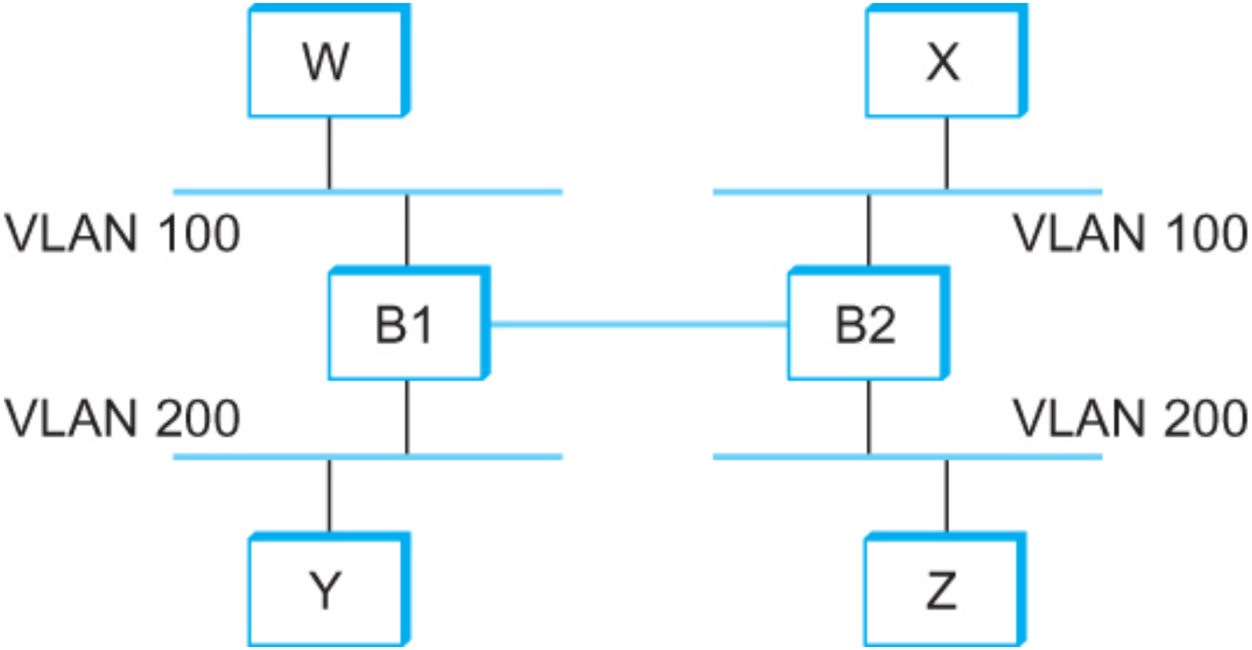
- Broadcast and Multicast
  - Forward all broadcast/multicast frames
    - Current practice
  - Learn when no group members downstream
  - Accomplished by having each member of group G send a frame to bridge multicast address with G in source field

Spanning Tree Algorithm

- Limitation of Bridges
  - Do not scale
    - Spanning tree algorithm does not scale
    - Broadcast does not scale
  - Do not accommodate heterogeneity

Spanning Tree Algorithm

- Virtual LAN


W
X
VLAN 100
VLAN 100
B1
B2
VLAN 200
VLAN 200
Y
Z