ECE 356/COMPSI 356
Computer Network Architecture

Ethernet Switches

Wednesday September 18, 2019

Recap

- Last lecture: switching technologies
- Material for this lecture: **PD 3.1.4**

Lecture Outline

- Ethernet switches, also knows as *bridges*
- Spanning tree algorithm
- Limitations of bridges & virtual LAN

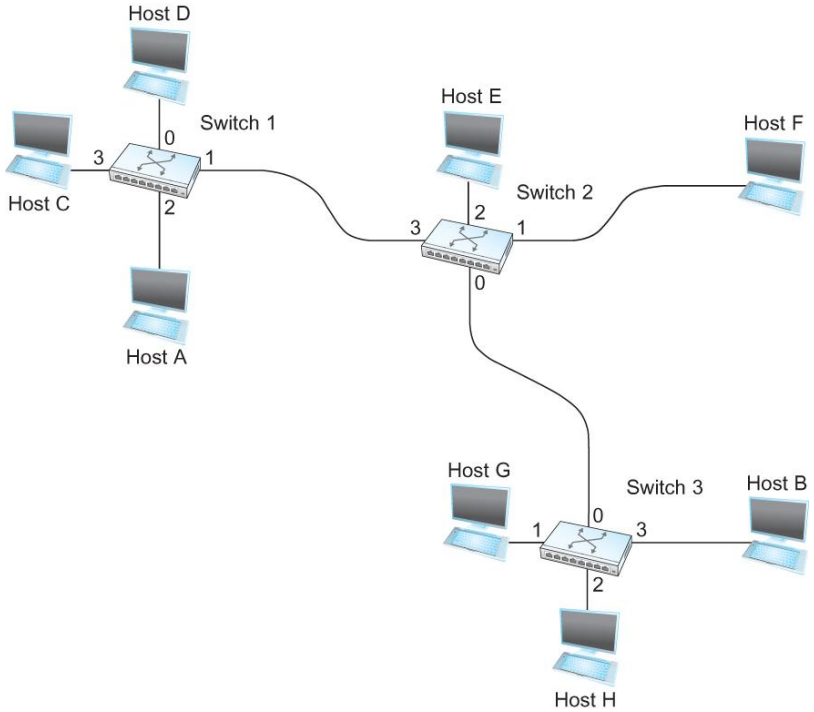
Ethernet Learning Bridges

- Local Area Network (LAN) switches
  - A specific type of a switch
  - Also called a *bridge*
    - Used to bridge different local area networks
- Overall design goal: **complete transparency**
  - "Plug-and-play"
  - Self-configuring without hardware or software changes
  - Bridges should not impact operations of existing LANs

Ethernet Bridges: Three Main Functions

- Forwarding of Ethernet frames
- Learning of addresses
- Spanning tree algorithm

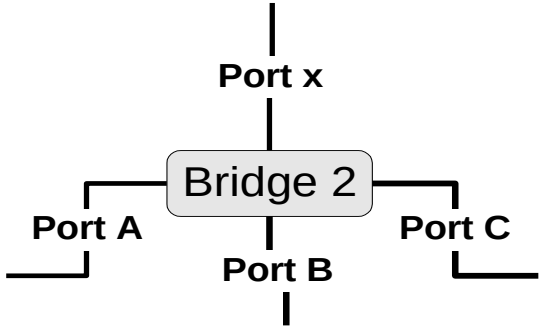
Recap: Switches Maintain a Forwarding Table



A	3
B	0
C	3
D	
E	
F	
G	
H	

Frame Forwarding

- Assume a MAC frame arrives on port *x*



Is MAC address of destination in forwarding table?

Found?

Not found ?

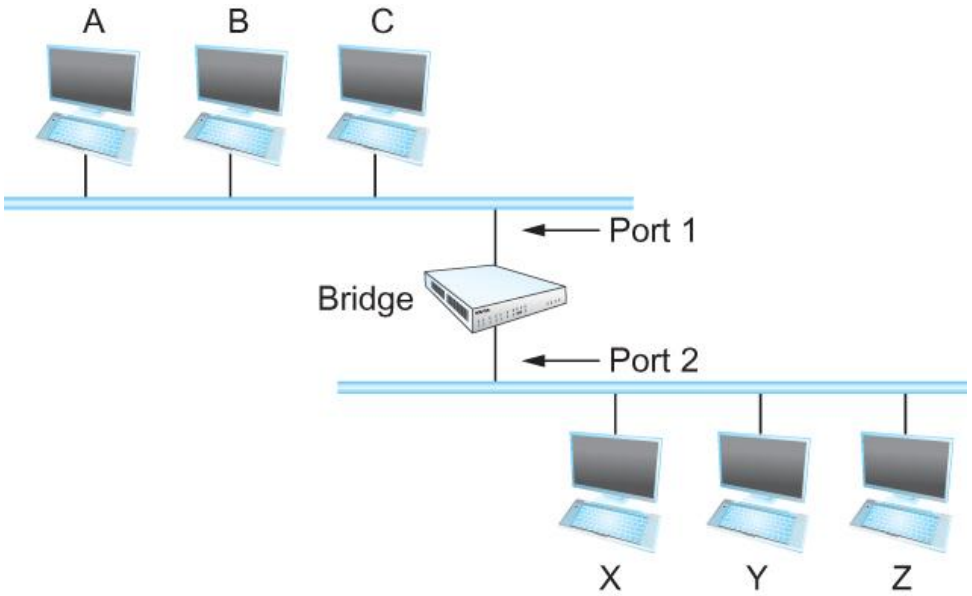
Forward the frame on the appropriate port

Flood the frame**,**
i.e., send the frame on all ports except port *x*

Address Learning

- When a bridge reboots, its forwarding table is empty
- Forwarding table entries are *learned* automatically with a simple heuristic:
  - The *source field* of a frame that arrives on a port tells which hosts are reachable from this port

Address Learning: An Example



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

Lecture Outline

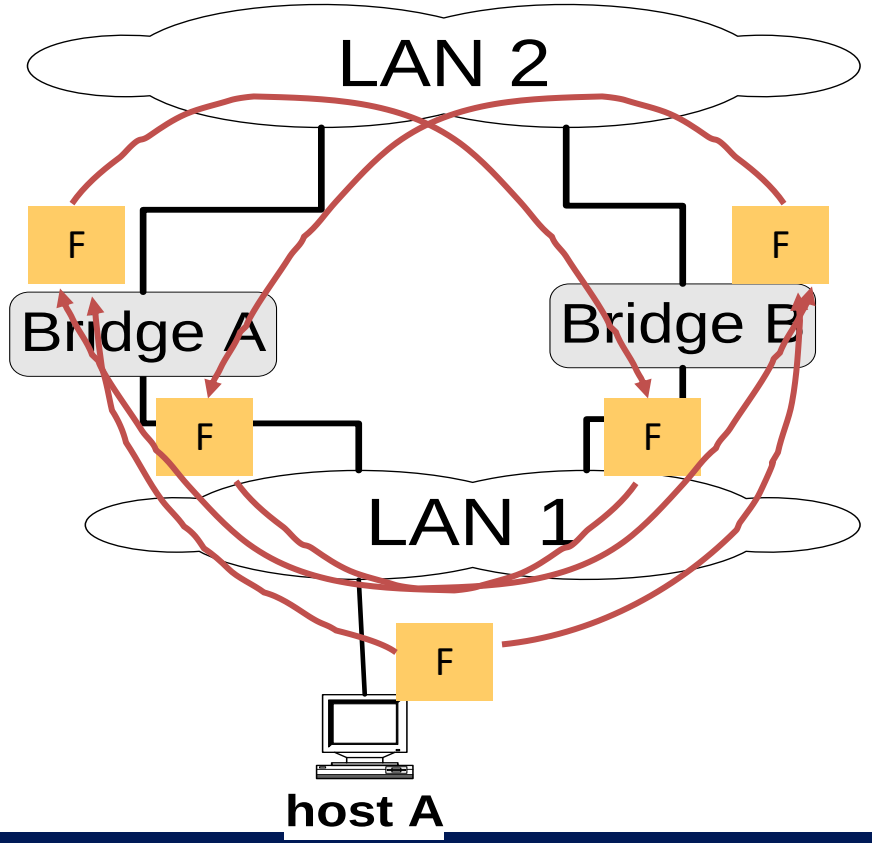
- Ethernet switches, also knows as *bridges*
- **Spanning tree algorithm**
- Limitations of bridges & virtual LAN

Danger of Loops

- Consider two LANs that are connected by two bridges
- Assume *host A* is transmitting a frame F with a broadcast address

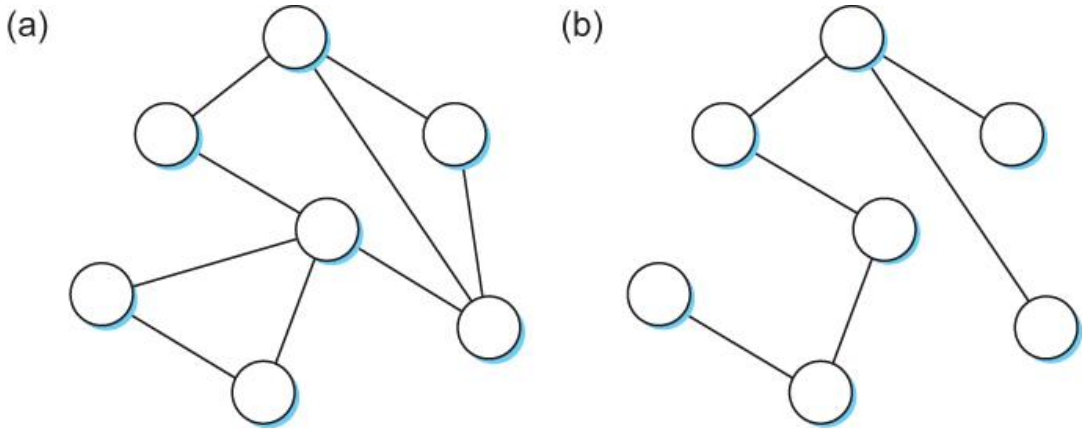
What is happening?

- Bridges A and B flood the frame to LAN 2
- Bridge B sees F on LAN 2, and updates the port mapping of MAC_A, and copies the frame back to LAN 1
- Bridge A does the same
- The copying continues



Spanning Tree Algorithm

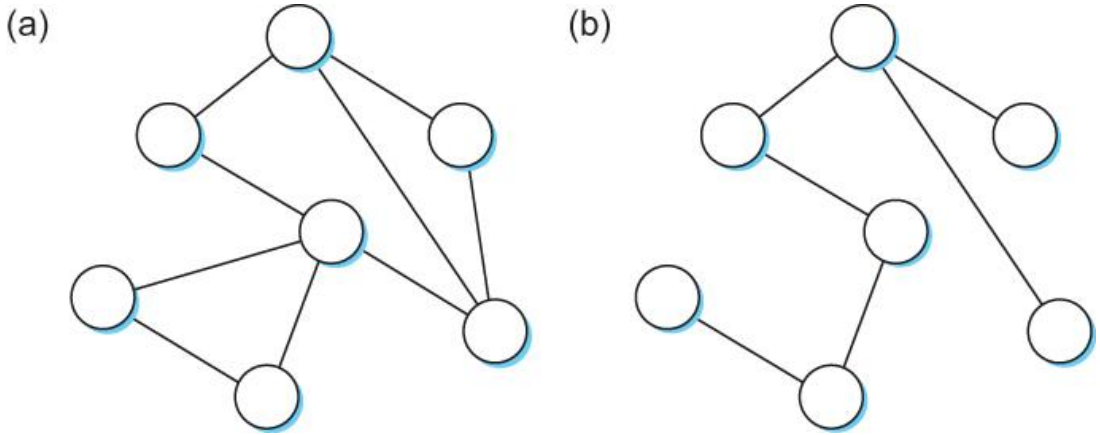
- A solution is the spanning tree algorithm that prevents loops in the topology
  - By Radia Perlman at DEC



Graph Theory on Spanning Trees

- For any connected graph consisting of nodes and edges connecting pairs of nodes, a spanning tree of edges maintains the connectivity of the graph but contains no loops
  - n-node graph, $n-1$ edges on a spanning tree
  - No redundancy

Can We Create Other Spanning Trees on This Graph?



Protocols vs. Algorithms

- Protocols are a set of rules that define message formats and actions to be taken when messages are sent or received
- Underlying a network protocol there is often a distributed algorithm
- Protocols must consider practical constraints, e.g.
  - Limited size of a field
  - Non-synchronized clocks

Algorhyme (the Spanning Tree Poem)

- 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 packets 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

The Protocol

- IEEE 802.1d has an algorithm that organizes the bridges as spanning tree in a dynamic environment
- Bridges exchange messages to configure the bridge (Configuration Bridge Protocol Data Unit, Configuration BPDUs) to build the tree
  - *Select ports they do and do not forward packets on*
  - Removing ports reduces a graph to an acyclic tree
  - A dynamic approach

What do the BPDUs do? (1/2)

- Elect a single bridge as the **root bridge**
  - Always forwards frames on all its ports
- Calculate the distance of the shortest path to the root bridge
- Each bridge can determine a **root port**, the port that gives the best path to the root

What do the BPDUs do? (2/2)

- Each LAN can determine a **designated bridge**, which is the bridge closest to the root
  - A LAN's designated bridge is the only bridge allowed to forward frames to and from the LAN for which it is the designated bridge
- LAN's **designated port** is the port that connects it to the designated bridge
- Select ports to be included in the spanning tree

Spanning Tree: Terms

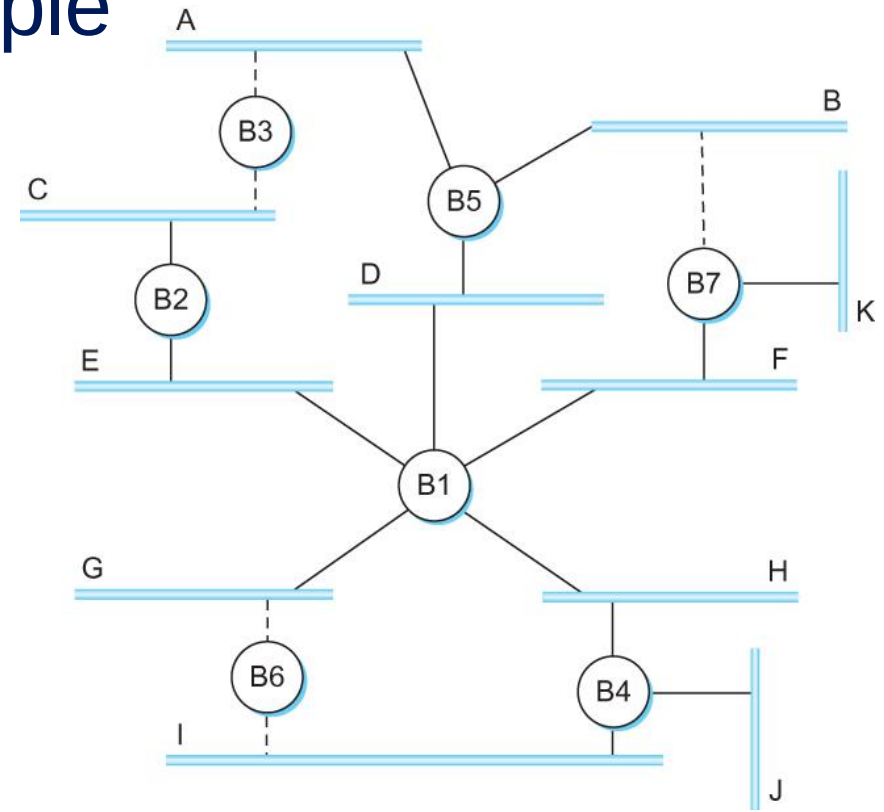
- Each bridge has a unique identifier: Bridge ID
  Bridge ID = {Priority: 2 bytes; Bridge MAC address: 6 bytes}
  - Priority is configured
  - Bridge MAC address is the lowest MAC addresses of all ports
- Each port within a bridge has a unique identifier (port ID)
- **Root Bridge:** The bridge with the lowest identifier is the root of the spanning tree

Spanning Tree: Terms

- **Root Path Cost**: For each bridge, the cost of the min-cost path to the root
  - Assume it is measured in #hops to the root
- **Designated Bridge, Designated Port:** Single bridge on a LAN that is closest to the root for this LAN
  - If two bridges have the same cost, select the one with the highest priority; if they have the same priority, select based on the bridge ID
  - If the min-cost bridge has two or more ports on the LAN, select the port with the lowest identifier

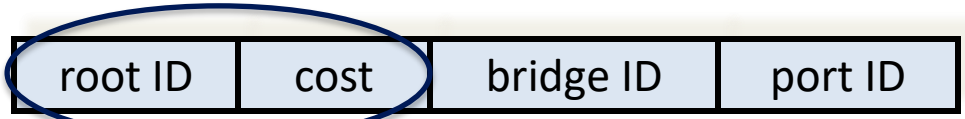
Spanning Tree: An Example

- B1 is a root bridge
- B5 is the designated bridge for LAN A
  - Shorter path to root than via B3
- B5 is the designated bridge for LAN B
  - Same distance as via B7, IDs used to break the tie



Spanning Tree Algorithm

- Each bridge is sending out BPDUs that contain the following information:



Root bridge (what the sender thinks it is)
Root path cost for sending bridge
Identifies sending bridge
Identifies the sending port

- The transmission of BPDUs results in the distributed computation of a spanning tree
- The convergence of the algorithm is very fast

Initializing the Spanning Tree Protocol

- Initially, all bridges assume they are the root bridge
- Each bridge B sends BPDUs of this form on its LANs from each port P:

- Each bridge looks at the BPDUs received on all its ports and its own transmitted BPDUs
- Root bridge is the one with the smallest received root ID that has been received so far
  - Whenever a smaller ID arrives, the root is updated

Ordering of Messages

- We define a *lexicographic ordering* of BPDU messages

M1

M2

M1 **advertises a better path** than M2 (“M1<M2”) if

(R1 < R2),

Or (R1 == R2) and (C1 < C2),

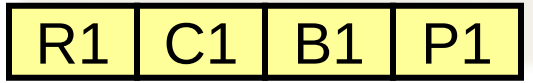
Or (R1 == R2) and (C1 == C2) and (B1 < B2),

Or (R1 == R2) and (C1 == C2) and (B1 == B2) and (P1 < P2)

Spanning Tree Protocol (1/2)

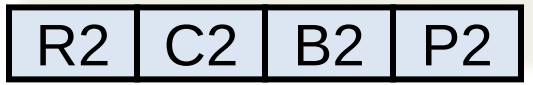
- Each bridge B looks on all its ports for BPDUs that are better than its own BPDUs
- Suppose a bridge with BPDU:

M1

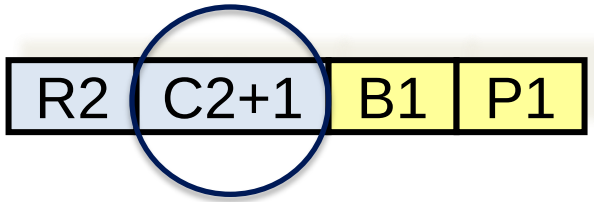


receives a “better” BPDU:

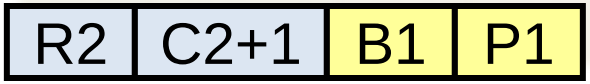
M2



- Then it will update the BPDU to:
  - Adds 1 to the distance to the root



Spanning Tree Protocol (2/2)



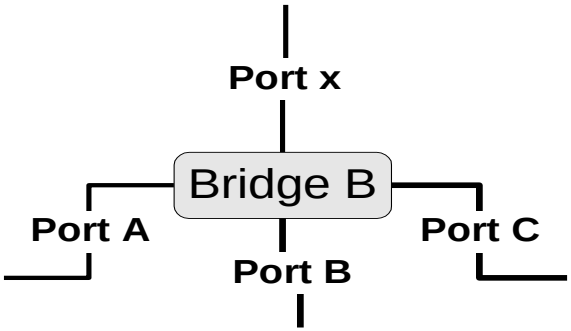
- However, the new BPDU is not necessarily sent out
- On each bridge, the port where the “best BPDU” (via relation “**<**“) was received is the **root port of the bridge**
  - No need to send out updated BPDUs on root port

When to End a BPDU

- Say, B has generated a BPDU for each port x

R	Cost	B	x

- B will send this BPDU on port x only if its BPDU is better (via relation “**<**“) than any BPDU that B received from port x
- In this case, B also assumes that it is the **designated bridge** for the LAN to which the port connects
- And port x is the **designated port** of that LAN



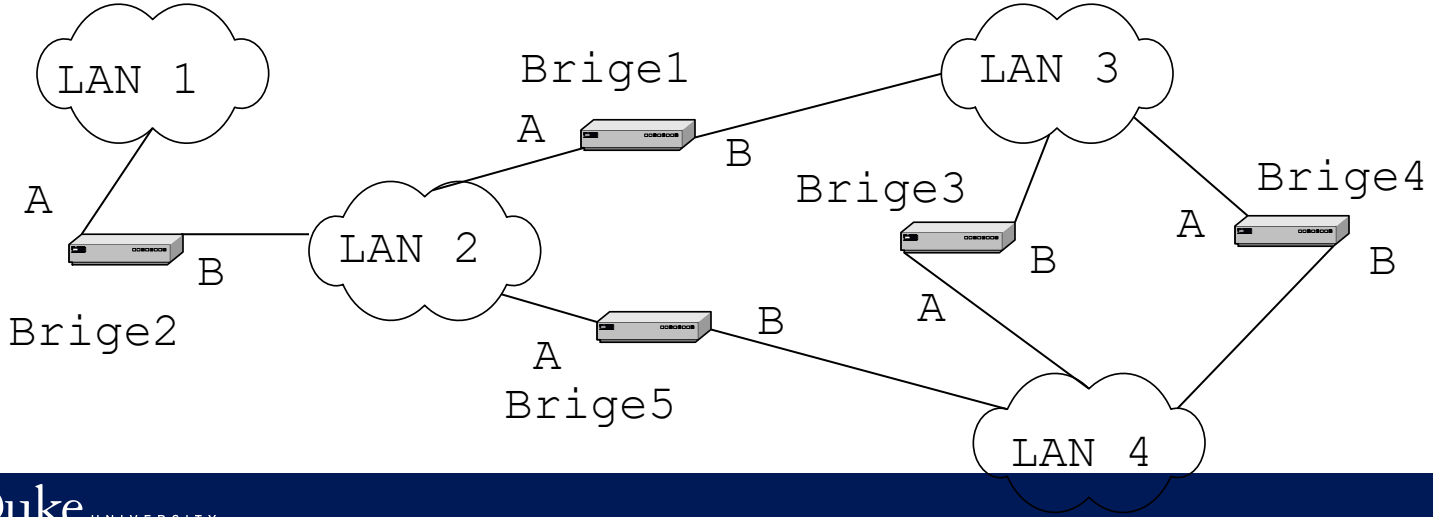
Selecting the Ports for the Spanning Tree

- Each bridge makes a local decision which of its ports are part of the spanning tree
- Now **B can decide which ports are in the spanning tree:**
  - B's root port is part of the spanning tree
  - All designated ports are part of the spanning tree
  - All other ports are not part of the spanning tree
- B's ports that are in the spanning tree will forward packets (=forwarding state)
- B's ports that are not in the spanning tree will not forward packets (=blocking state)

Example

root ID	cost	bridge ID	port ID

- All bridges send out their BPDU's once per second
- All bridges send their BPDUs at the same time
- Bridge1 < Bridge2 < Bridge3 < Bridge4 < Bridge5
- All bridges are turned on simultaneously at time T=0 sec.



Example: BPDUs Sent

	Bridge1	Bridge2	Bridge3	Bridge4	Bridge5
T=1sec	Send: A: (B1,0,B1,A) B: (B1,0,B1,B) Recv: A: (B5,0,B5,A) (B2,0,B2,B) B: (B3,0,B3,B) (B4,0,B4,A)	Send: A: (B2,0,B2,A) B: (B2,0,B2,B) Recv: A: B: (B1,0,B1,A) (B5,0,B5,A)	Send: A:(B3,0,B3,A) B:(B3,0,B3,B) Recv: A: (B5,0,B5,B) (B4,0,B4,B) B: (B1,0,B1,B) (B4,0,B4,A)	Send: A:(B4,0,B4,A) B:(B4,0,B4,B) Recv: A: (B3,0,B3,B) (B1,0,B1,B) B: (B3,0,B3,A) (B5,0,B5,B)	Send: A:(B5,0,B5,A) B:(B5,0,B5,B) Recv: A: (B2,0,B2,B) (B1,0,B1,A) B: (B3,0,B3,A) (B4,0,B4,B)

Example: BPDUs Sent

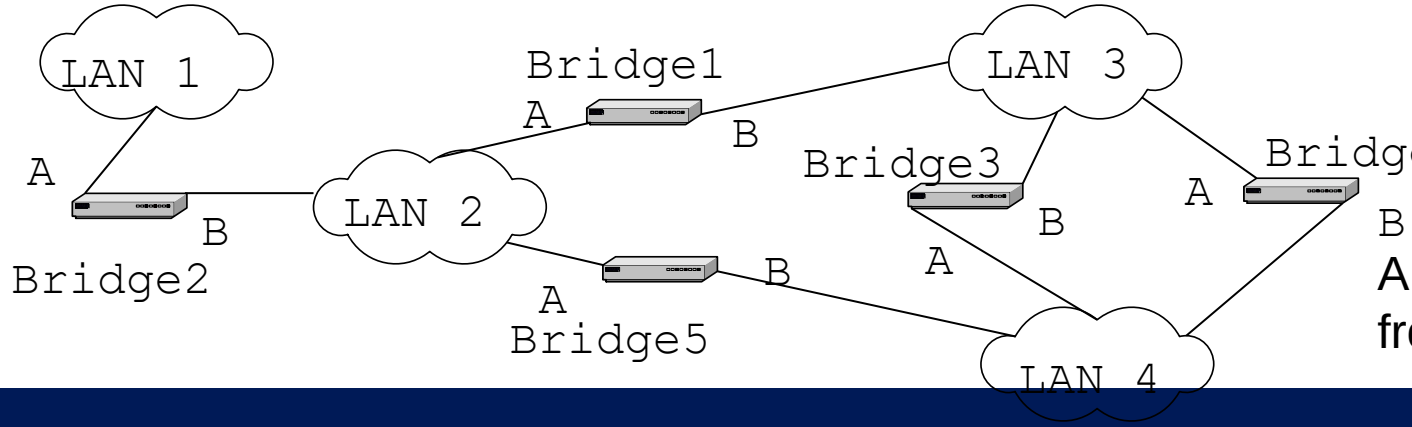
	Bridge1	Bridge2	Bridge3	Bridge4	Bridge5
T=2sec	D-port: A,B Send: A: (B1,0,B1,A) B: (B1,0,B1,B) Recv:	R-port: B D-port: A Send: A: (B1,1,B2,A) Recv: A: B: (B1,0,B1,A)	R-port: B D-port: A Send: A: (B1,1,B3,A) Recv: A: (B1,1,B4,B) (B1,1,B5,B) B: (B1,0,B1,B)	R-port: A D-port: B Send: B: (B1,1,B4,B) Recv: A: (B1,0,B1,B) B: (B1,1,B3,A) (B1,1,B5,B)	R-port: A D-port: B Send: B: (B1,1,B5,B) Recv: A: (B1,0,B1,A) B: (B1,1,B3,A) (B1,1,B4,B)

Example: BPDUs Sent

	Bridge 1	Bridge 2	Bridge 3	Bridge4	Bridge5
T=3sec	D-port: A,B Send: A: (B1,0,B1,A) B: (B1,0,B1,B) Recv:	R-port: B D-port: A Send: A: (B1,1,B2,A) Recv: A: B: (B1,0,B1,A)	R-port: B D-port: A Send: A: (B1,1,B3,A) Recv: A: B: (B1,0,B1,B)	R-port: A Blocked: B Recv: A: (B1,0,B1,B) B: (B1,1,B3,A)	R-port: A Blocked: B Recv: A: (B1,0,B1,A) B: (B1,1,B3,A)

Example: The Spanning Tree

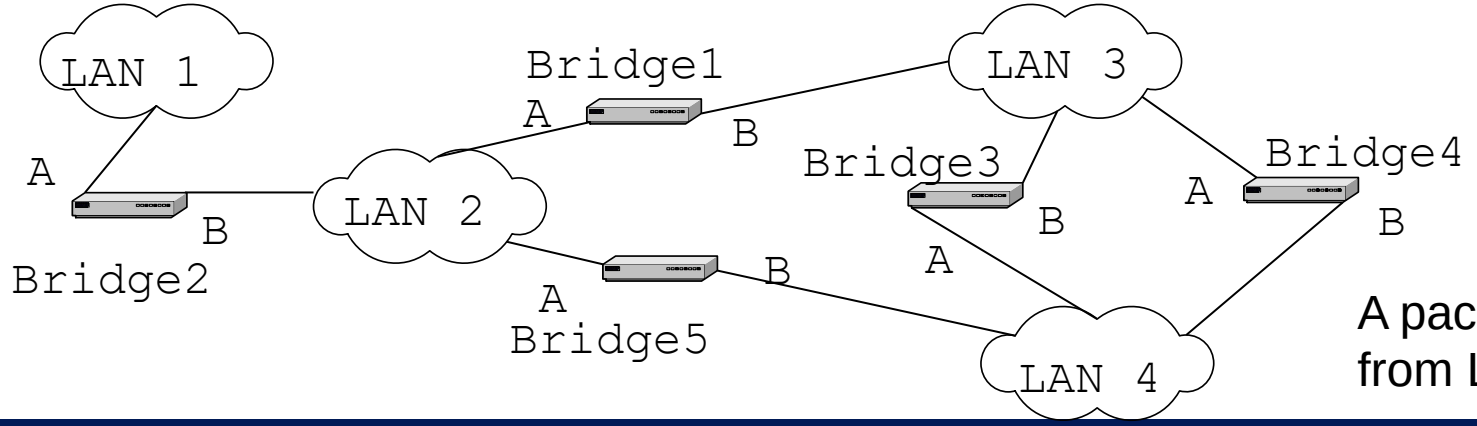
	Bridge1	Bridge2	Bridge3	Bridge4	Bridge5
Root Port					
Designated bridge					
Designated ports					



A packet is sent from LAN2

Example: The Spanning Tree

	Bridge1	Bridge2	Bridge3	Bridge4	Bridge5
Root Port		B	B	A	A
Designated bridge	LAN2,3	LAN1	LAN4		
Designated ports	A,B	A	A		



A packet is sent from LAN2

Lecture Outline

- Ethernet switches, also knows as *bridges*
  - Forwarding
  - Address learning
  - Spanning tree algorithm
- **Limitations of bridges & virtual LAN**

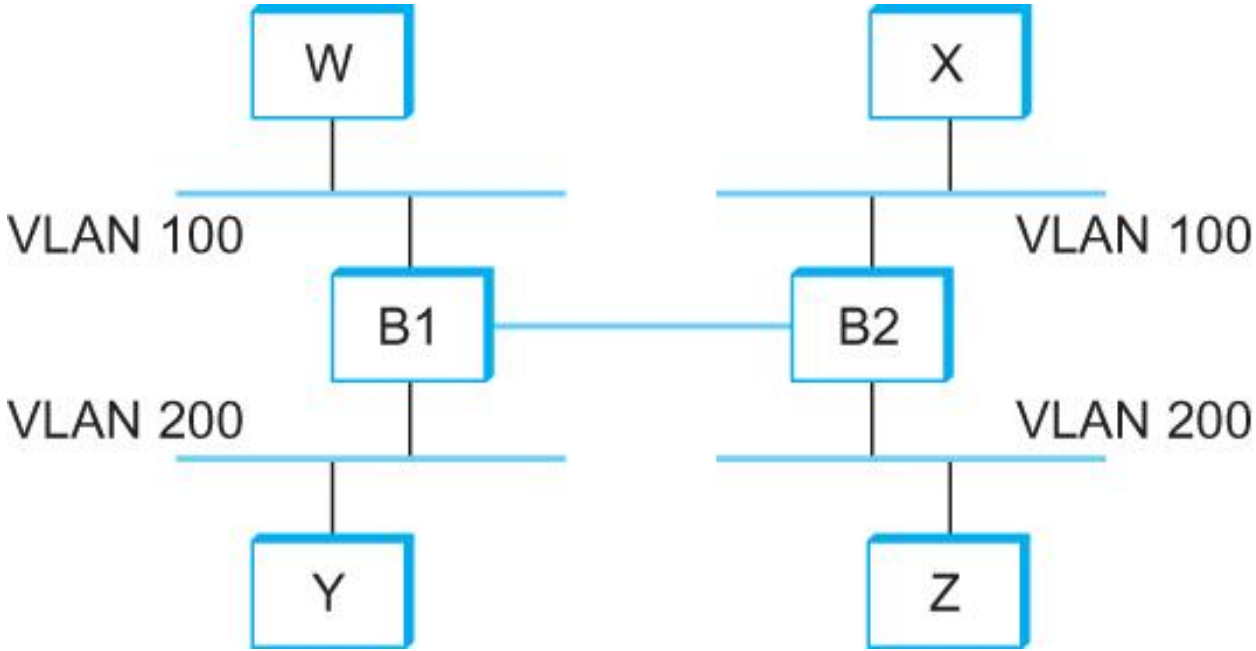
Limitations of Bridges

- Scalability – only 10s of LANs can be connected in practice
  - Spanning tree algorithms scales linearly
  - Broadcast packets reach *every host*
- Security
  - Every host can snoop
- Non-heterogeneity
  - Can't connect ATM networks

Virtual LANs (VLANs)

- To address the scalability and security issues
- A bridge's port is configured to have a VLAN ID
- Each VLAN has a spanning tree
- A VLAN header is inserted to a packet
- Packets are flooded to ports with the same VLAN ID

VLAN: An Example



Lecture Summary

- Ethernet switches, also knows as *bridges*
  - Forwarding
  - Address learning
  - Spanning tree algorithm
- Limitations of bridges & virtual LAN

Next Lecture

- IP protocol