

Lecture 7: Bridging

CSE 123: Computer Networks
Alex C. Snoeren



Lecture 7 Overview



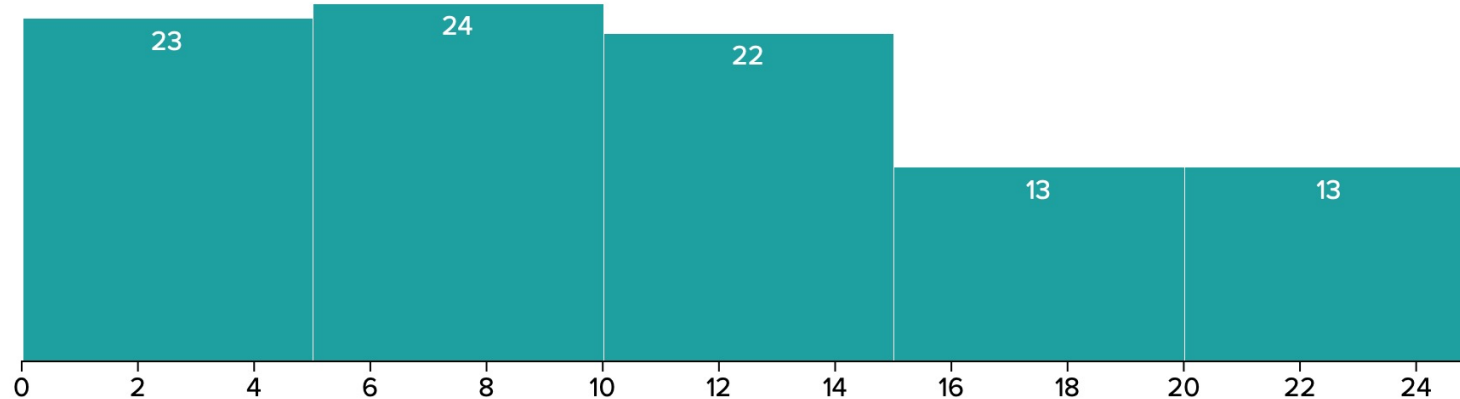
- TCP connection termination
- Connecting links
 - ◆ Forwarding
 - ◆ Collision domains
- Bridging
 - ◆ Store and forward
 - ◆ Learning bridges

Project 1a



Review Grades for **Project 1a**

● GRADES PUBLISHED



MINIMUM

0.0

MEDIAN

10.0

MAXIMUM

25.0

MEAN

10.6

STD DEV

7.54

Connection Teardown



- TCP provides a “symmetric” close
 - ◆ Both sides shutdown independently

Why a symmetric close?

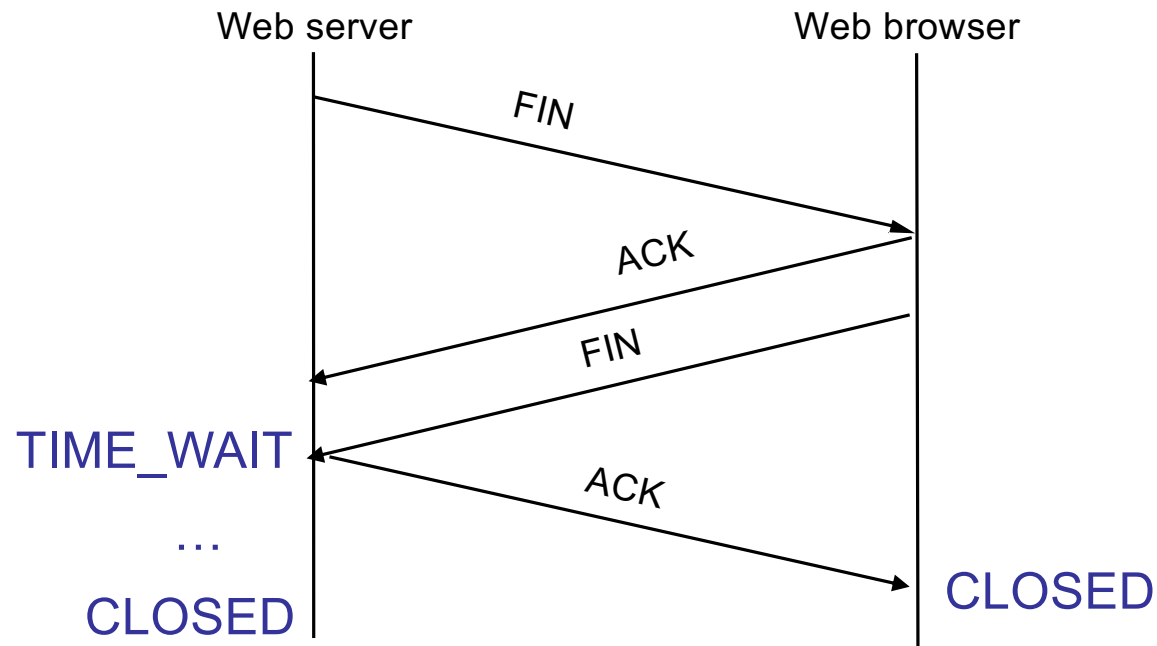
- A. Need to ensure no data loss**
- B. Data transfer is bi-directional**
- C. Both ends need to know when they can “forget” about connection**
- D. All of the above**

Connection Teardown



- TCP provides a “symmetric” close
 - ◆ Both sides shutdown independently
- Orderly release by sender and receiver when done
 - ◆ Delivers all pending data and “hangs up”
- Cleans up state in sender and receiver

TCP Connection Teardown





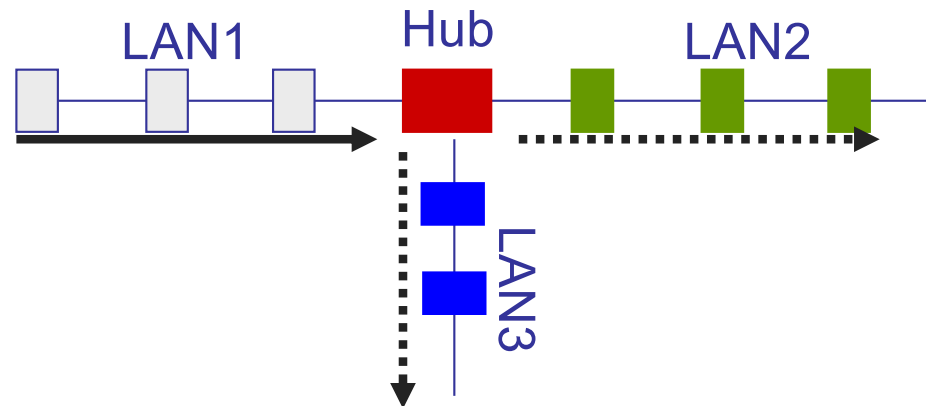
One link is not enough

- Often called a **Local Area Network (LAN)**
 - ◆ Link is multiplexed across time
 - ◆ Frames have a link-layer header with addresses of source and destination of hosts on the LAN
- One shared link (a **bus**) limits scale in terms of:
 - ◆ Distance (e.g., 2500 m for Ethernet)
 - ◆ Performance (Capacity shared across all nodes)
- A better alternative is to have multiple busses
 - ◆ Each bus is of a limited size, scale, number of hosts, etc.
 - ◆ Need the ability to connect multiple busses together



Hubs/Repeaters

- Physical layer device
 - ◆ One “port” for each LAN
 - ◆ Repeat received *bits* on one port out *all* other ports



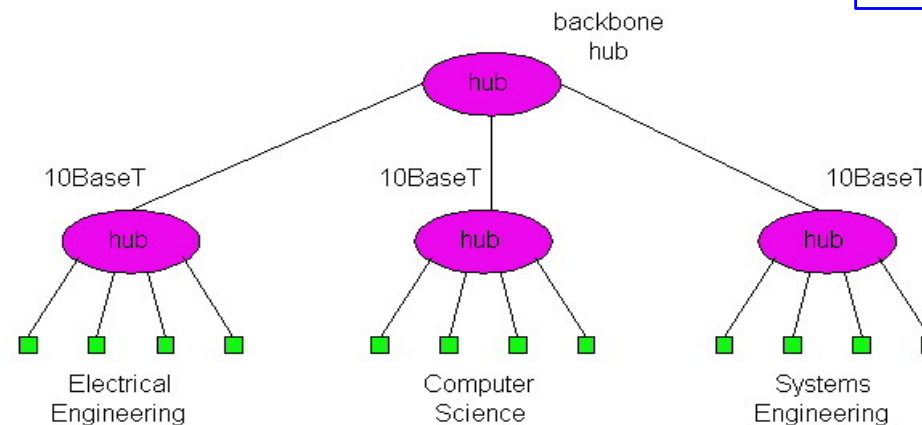


Hub Advantages

- Hubs can be arranged into hierarchies
 - ♦ Ethernet: up to four hubs between any pair of nodes
- Most of LAN continues to operate if “leaf” hub dies
- Simple, cheap

If a host in EE sends a frame, who receives it?

- A. Hosts in ECE
- B. Hosts in CSE
- C. Hosts in SE
- D. All of the above





Still One Big Bus

- Single **collision domain**
 - ♦ No improvement in max throughput
 - ♦ Average throughput < as # of nodes increases
- Still limited in distance and number of hosts
 - ♦ Collision detection requirements
 - ♦ Synchronization requirements
- Requires performance homogeneity
 - ♦ Can't connect 1 Gbps and 100 Gbps networks

Bridges

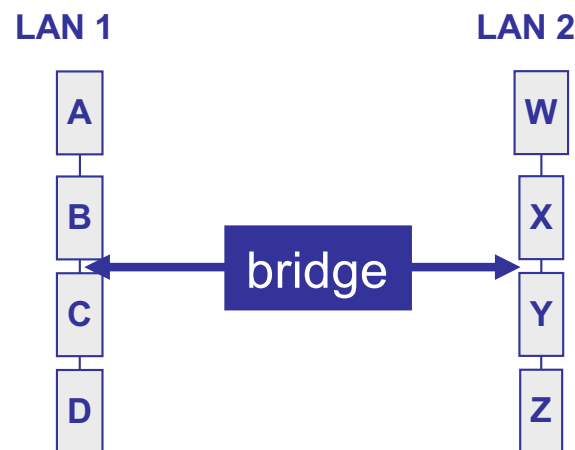


- Store and forward device
 - ◆ Data-link layer device
 - ◆ Buffers entire packet and *then* rebroadcasts it on other ports
- Creates *separate* collision domains
 - ◆ Uses link-layer protocol for access to each LAN (i.e., acts like a host on that LAN)
 - ◆ Can accommodate different speed interfaces (issues?)
 - ◆ Separate CDs improves throughput (why?)
- Can significantly improve performance
 - ◆ Not all frames go everywhere. (Why did they with a hub?)

Selective Forwarding



- Only rebroadcast a frame to the LAN where its destination resides
 - ♦ If A sends packet to X, then bridge must forward frame
 - ♦ If A sends packet to B, then bridge shouldn't



Who should receive a frame sent from A to Z?

- A. Only Z
- B. W, X, Y, and Z
- C. B, C, D, and Z
- D. B, C, D, W, X, Y, and Z



Forwarding Tables

- Need to know “destination” of frame
 - ♦ Destination address in frame header (48bit in Ethernet)
- Need know which destinations are on which LANs
 - ♦ One approach: statically configured by hand
 - » Table, mapping address to output port (i.e. LAN)
 - ♦ But we’d prefer something automatic and dynamic...
- Simple algorithm:

```
Receive frame f on port q
Lookup f.dest for output port /* know where to send it? */
If f.dest found
then if output port is q then drop /* already delivered */
    else forward f on output port;
else flood f;
/* forward on all ports but the one where frame arrived*/
```

Host	Port
A	1
B	1
C	1
D	1
W	2
X	2
Y	3
Z	2

Learning Bridges



- Eliminate manual configuration by learning which addresses are on which LANs

- Basic approach
 - ◆ If a frame arrives on a port, then associate its **source** address with that port
 - ◆ As each host transmits, the table becomes accurate
- What if a node moves? Table aging
 - ◆ Associate a timestamp with each table entry
 - ◆ Refresh timestamp for each new packet with same source
 - ◆ If entry gets too stale, remove it

Host	Port
A	1
B	1
C	1
D	1
W	2
X	2
Y	3
Z	2

For next time...



- Keep reading 3 – 3.1
- Figure out what went wrong with 1a code
- Get help with 1b NOW!