CSCI-1680

Building Links and (Local) Networks

Nick DeMarinis

Based partly on lecture notes by Rodrigo Fonseca, David Mazières, Phil Levis, John Jannotti

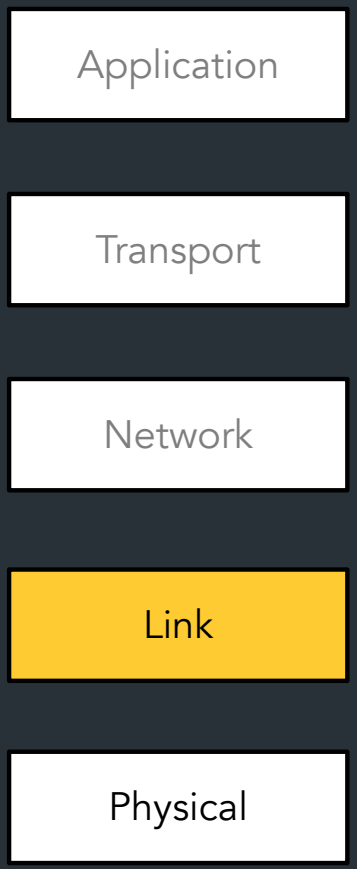
Today

Last time: how to send over a link

Today: how to build a network with links?

- Sharing links
- Case study: Ethernet (and Wifi)
  - Network interfaces: How you interact with the link layer
- How switching works

What does "link layer" mean?



Service: user-facing application.
Application-defined messages

"MESSAGE"

Service: multiplexing applications
Reliable byte stream to other node (TCP),
Unreliable datagram (UDP)

Service: move packets to any other node in the network
Internet Protocol (IP)

"PACKET"

L2

Service: move frames to other node across link.
May add reliability, medium access control

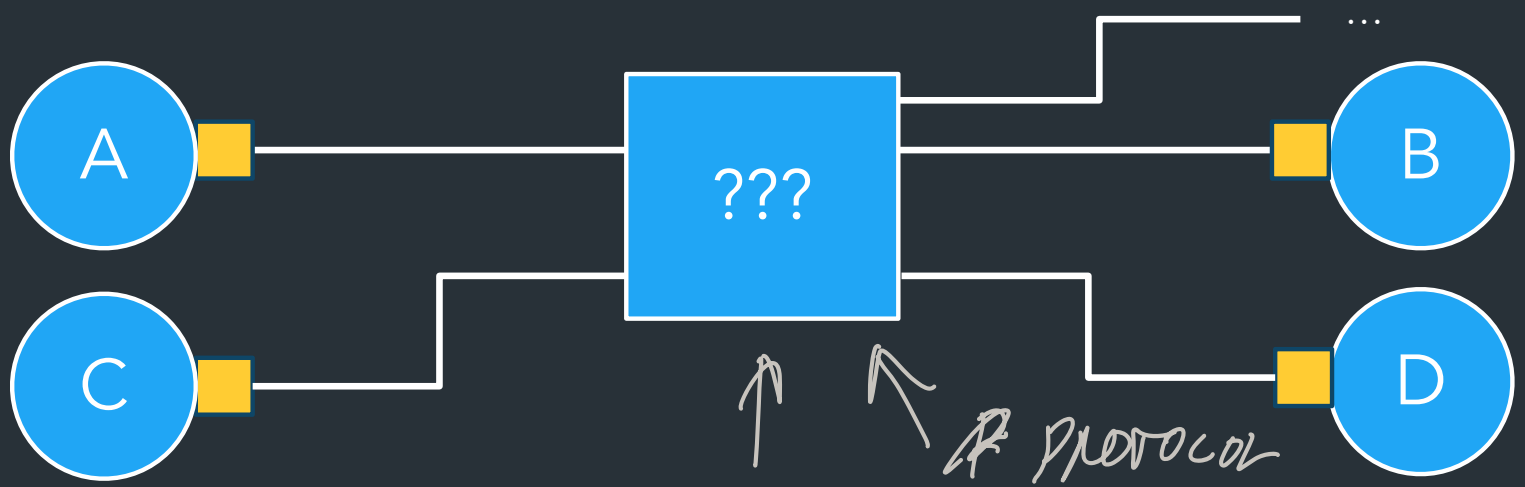
"FRAME"

Service: move bits to other node across link

The main idea

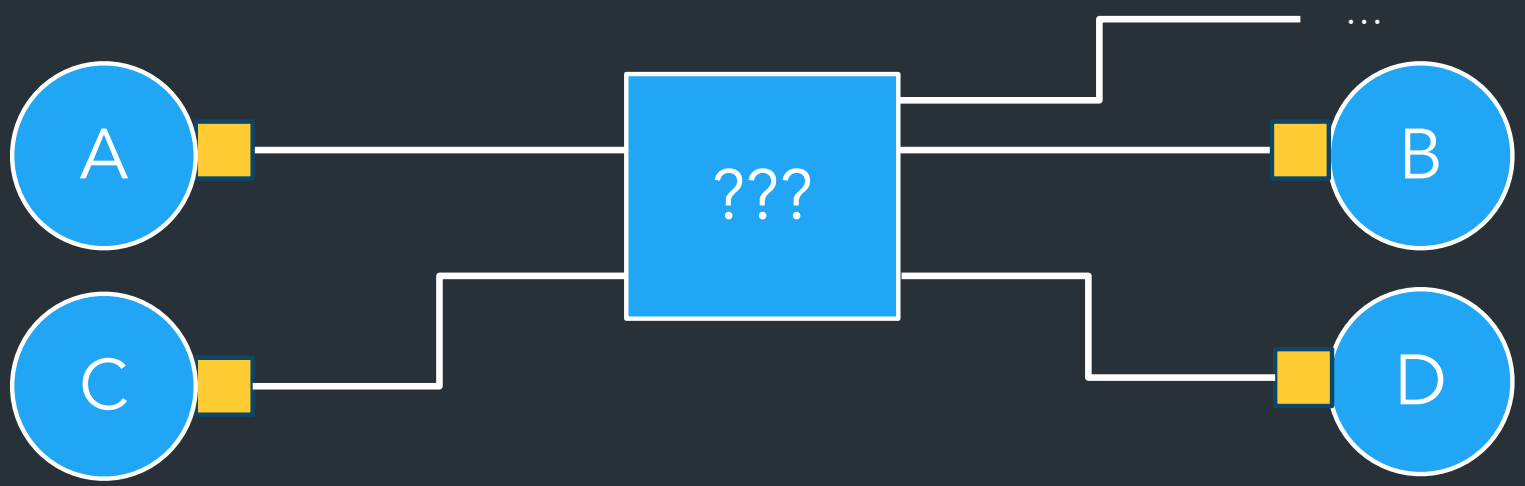
Sending bits over a channel….

What does “link layer” mean?



- Multiple hosts => shared channel
- Need ways to allow “small” number of hosts to communicate

What does “link layer” mean?



- Multiple hosts => shared channel
- Need ways to allow “small” number of hosts to communicate

“Small” => Within a building, floor of office, etc

Related term: Local Area Network (LAN)

HOME OFFICE

How to share the channel?

Medium Access Control (MAC)

↳ Protocol for how to "talk" on channel

Medium Access Control

Idea: Control access to shared physical medium

=> No more than one device can be "talking" at one time

Need a protocol for "who can talk when?"

WIRE
FREQUENCY

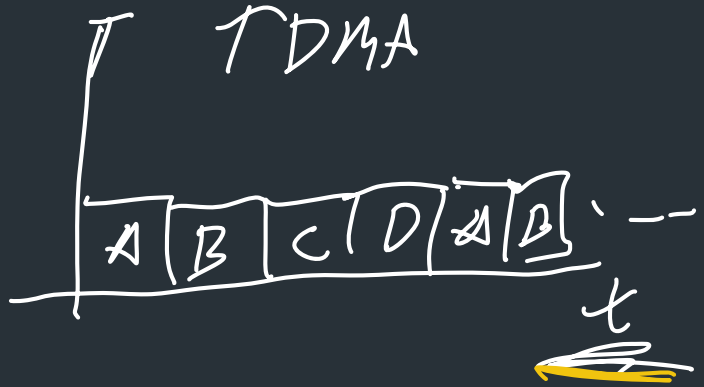
An example of multiplexing => sharing the channel among multiple devices

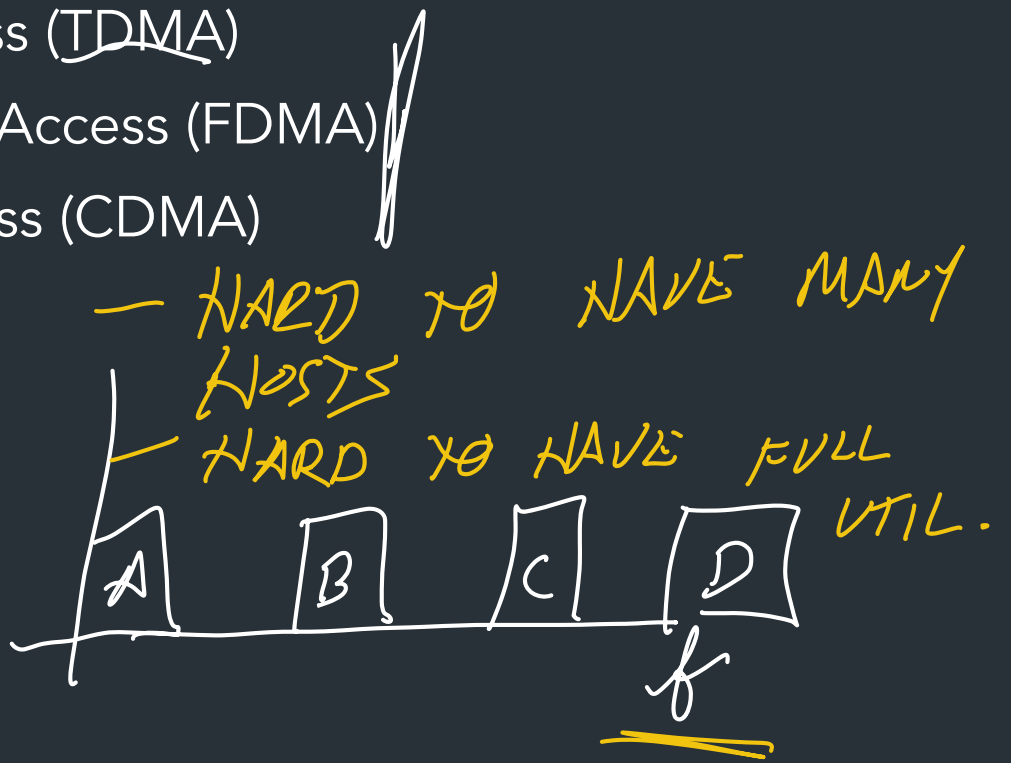
High-level: MAC approaches

Partitioned Access: divide the channel into fixed slots

- Time Division Multiple Access (TDMA)
- Frequency Division Multiple Access (FDMA)
- Code Division Multiple Access (CDMA)

Problems?





High-level: MAC approaches

Partitioned Access: divide the channel into fixed slots

- Time Division Multiple Access (TDMA)
- Frequency Division Multiple Access (FDMA)
- Code Division Multiple Access (CDMA)

Problems?

⇒ **Hard to maximize channel utilization (eg. what happens if only one person is talking?)**

High-level: MAC approaches

Random Access: no fixed slots: "ask" to talk, or just talk and hope for the best

- Carrier Sense Multiple Access / Collision Detection (CSMA/CD)
- Carrier Sense Multiple Access / Collision Avoidance (CSMA/CA)
- RTS/CTS (Request to Send/Clear to Send)
- Token-based

(KARE)

Problems?

COULD

- RETRY
- WAIT LONGER
- ASK IF OKAY TO TRANSMIT

High-level: MAC approaches

Random Access: no fixed slots: "ask" to talk, or just talk and hope for the best

- Carrier Sense Multiple Access / Collision Detection (CSMA/CD)
- Carrier Sense Multiple Access / Collision Avoidance (CSMA/CA)
- RTS/CTS (Request to Send/Clear to Send)
- Token-based

- WAY TO MINIMIZE COLLISIONS
- WAY TO HAVE "FAIRNESS"

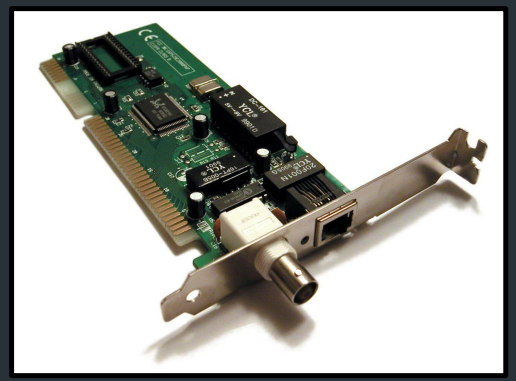
Problems?

⇒ **Hard to maintain "fairness"**
(eg. one host dominating channel)

Interface: device that connects something to a network

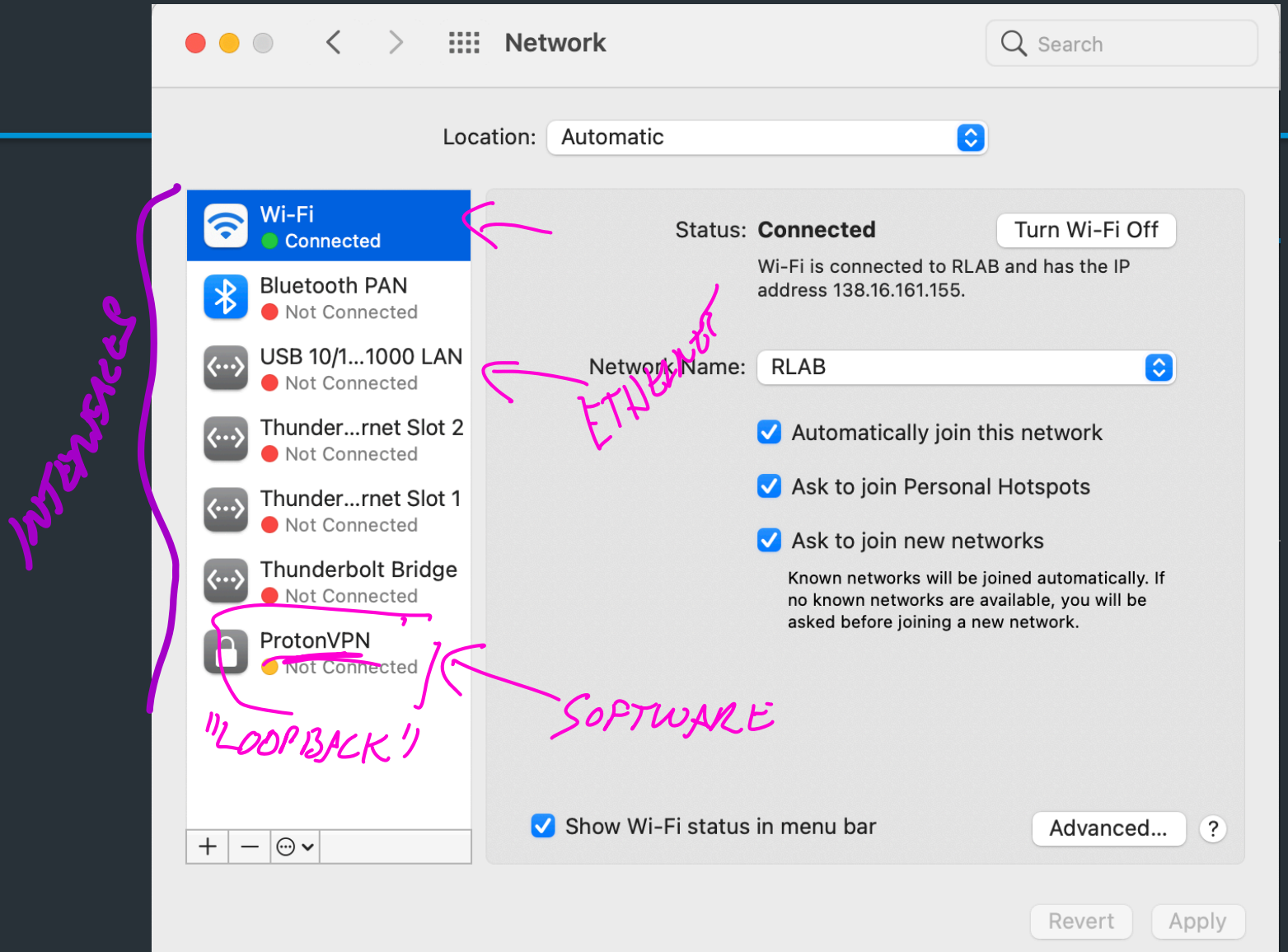
- OS abstraction for a network device
- Physical hardware that does the "talking"

  => Network Interface Card (NIC)



Common interfaces

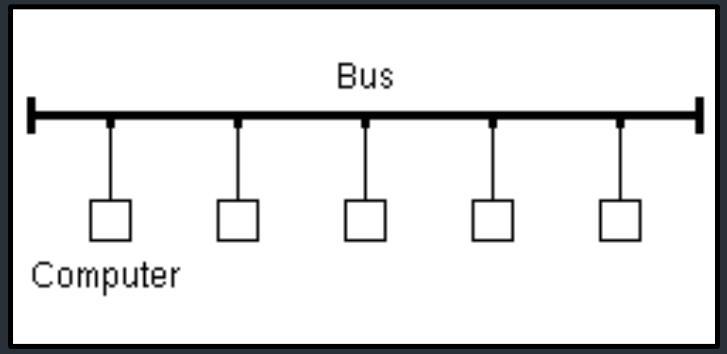
- Loopback: Virtual, only for local host
- Wifi, Ethernet, Bluetooth, …

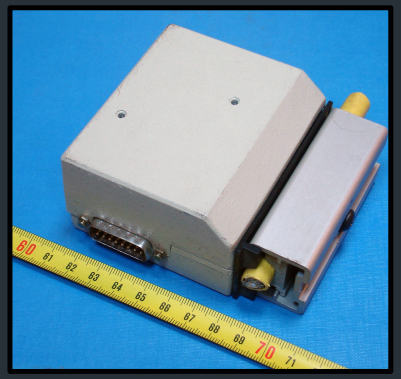

Network
Search
Location: Automatic
Wi-Fi
Connected
Bluetooth PAN
Not Connected
USB 10/1...1000 LAN
Not Connected
Thunder...rnet Slot 2
Not Connected
Thunder...rnet Slot 1
Not Connected
Thunderbolt Bridge
Not Connected
ProtonVPN
Not Connected
Status: Connected
Turn Wi-Fi Off
Wi-Fi is connected to RLAB and has the IP address 138.16.161.155.
Network Name: RLAB
Automatically join this network
Ask to join Personal Hotspots
Ask to join new networks
Known networks will be joined automatically. If no known networks are available, you will be asked before joining a new network.
Show Wi-Fi status in menu bar
Advanced...
?
Revert
Apply
INTERFACES
ETHERNET
SOFTWARE
"LOOPBACK"

Example: Ethernet

Ethernet's evolution

Originally, a shared medium with all hosts





- Basic idea: all hosts can see all frames, read a frame if it matches your hardware address
- Implications?

=>Can have collisons!

Classical Ethernet: Problems

- Problem: shared medium, all hosts in the same "collision domain"

ALL HOSTS COULD CAUSE COLLISIONS for/ OTHERS.

Classical Ethernet: Problems

- Problem: shared medium, all hosts in the same "collision domain"
- Transmit algorithm
  - If line is idle, transmit immediately
  - Upper bound message size of 1500 bytes
  - If line is busy: wait until idle and transmit immediately
- Generally possible to detect collisions, deal with it

CAN DETECT A COLLISION

CSMA/CD: Carrier Sense Multiple Access / Collision Detection

When to transmit again?

- Delay and try again: exponential backoff
- nth time: $k \times 51.2\mu s$, for $k = U\{0..(2^{\min(n,10)}-1)\}$
  - 1st time: 0 or 51.2μs
  - 2nd time: 0, 51.2, 102.4, or 153.6μs
- Give up after several times (usually 16)

- Exponential backoff is a useful, general technique

Does this scale?

Ethernet Recap

- Service provided: send frames among stations with specific addresses
- All nodes in the same "collision domain"

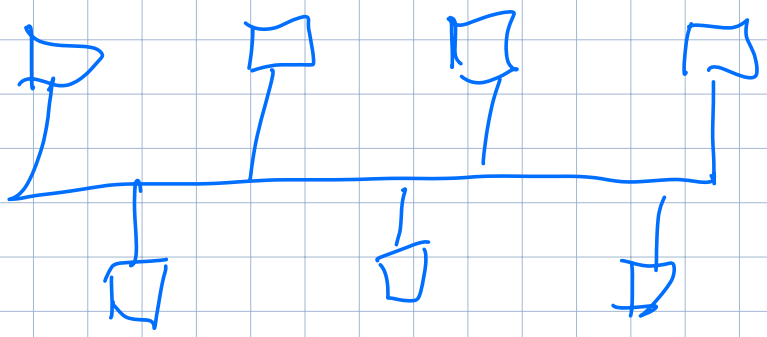
↳ IN MODERN ETHERNET, NOT SO MUCH A PROBLEM.

Avoiding collisions

- Early method: bridging

EARLY ETHERNET:

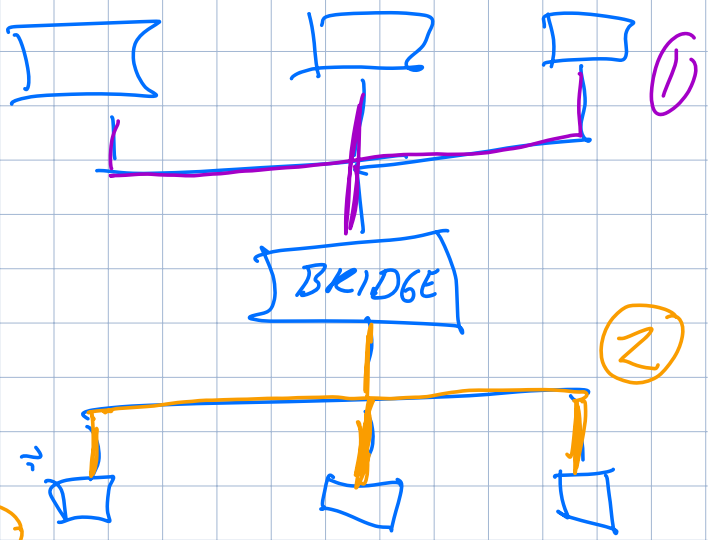
ALL HOSTS ON SAME COLLISION DOMAIN



- DOESN'T SCALE

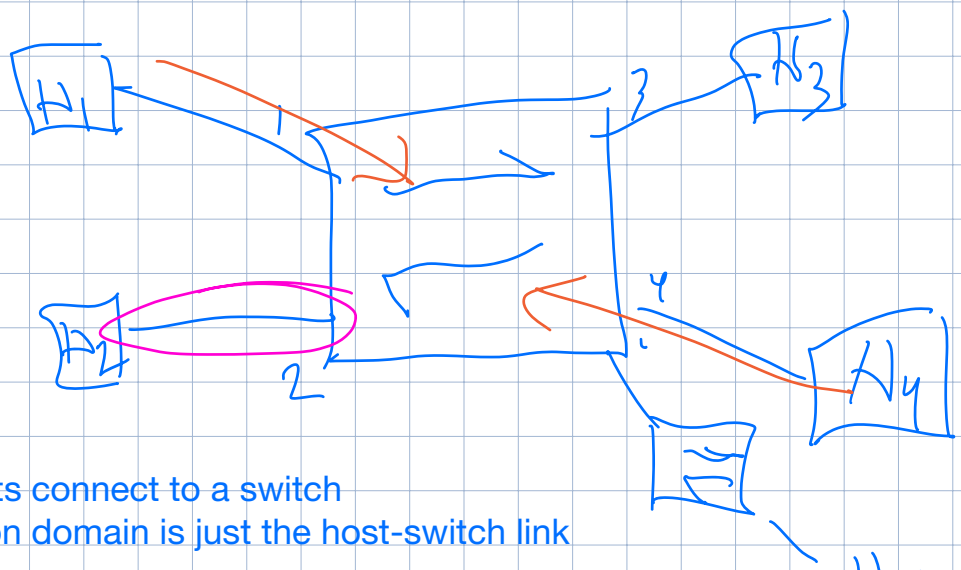
- WANT ABILITY TO SEPARATE COLLISION DOMAIN WHILE STILL BEING ABLE TO TALK TO EVERYONE.

EARLY WAY: BRIDGES
- CONNECT TWO NETWORKS
- ONLY FORWARD BETWEEN ① ↔ ② IF NECESSARY



HERE: ONE LAN, 2 COLLISION DOMAINS ① ②

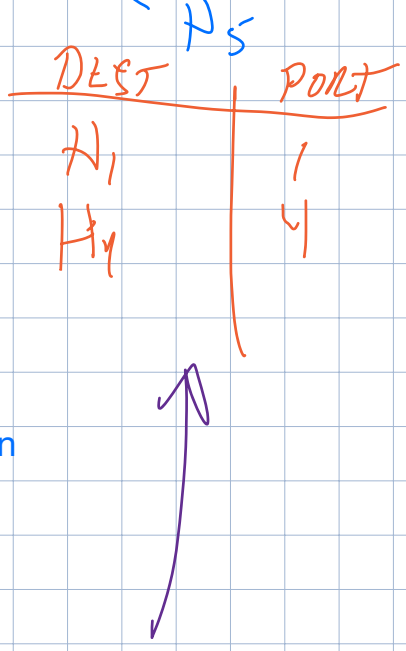
Modern way: an Ethernet switch



All hosts connect to a switch
Collision domain is just the host-switch link

=> In modern times, both sides of one link can transmit/receive at the same time (full duplex)
=> Really hard to have a collision

Switch receives frames, looks at destination address
- Decides which port to send packet to
- Queue up packets until the destination port is idle
=> Lots of engineering involved in how to design a switch to do this quickly, queue efficiently



MAC learning: as packets are sent, switch builds a table of mac addresses it has seen before, remembers the port used

(More info in next couple of pages)

Modern way: switching

Switch: network device that forwards frames (packets) between ports

- All hosts connect to a switch
- Collision domain is host-switch
- Switch buffers packets, forwards to destination when its port is idle

How to know which devices is on which port?

MAC Learning

- Switches “learn” which host lives on which port by watching traffic
- If you don’t know, flood to all ports!

MAC Learning

- Switches "learn" which host lives on which port by watching traffic
- If you don't know, flood to all ports!

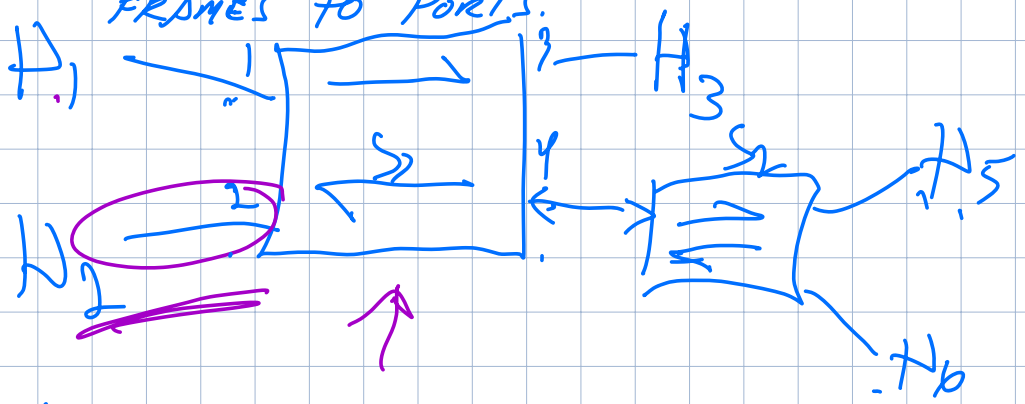
SOME SECURITY PROBLEMS
- CAN "SPOOF" (FORGE) ADDRESS
- CAN FILL TABLE.

REALLY JUST A CACHE!

MAC learning is just an optimization vs. old version (but a pretty good one…)

MODERN ETHERNET: SWITCHES

SWITCH: NETWORK DEVICE THAT FORWARDS FRAMES TO PORTS.

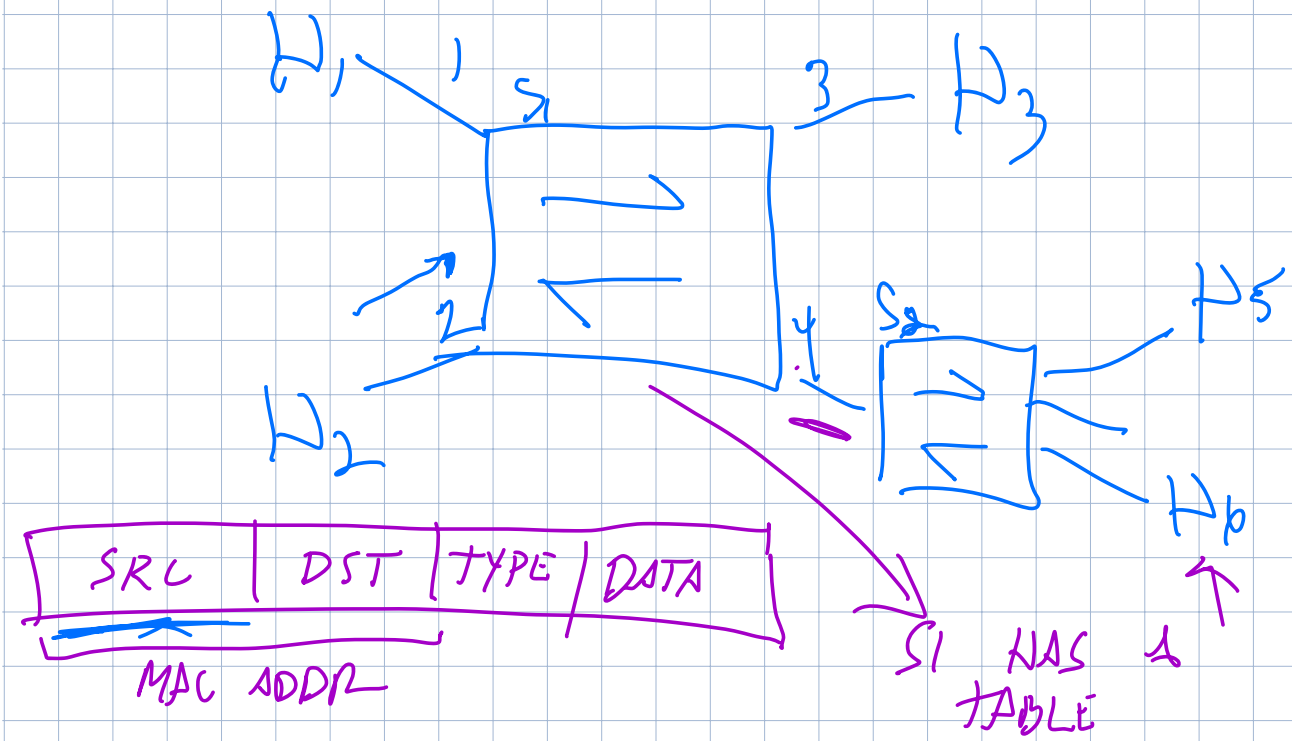


- ALL HOSTS CONNECT TO A SWITCH

- COLLISION DOMAIN IS JUST THE LINK OF HOST – SWITCH.
  - IN MODERN TIMES, BOTH SIDES OF A LINK CAN TRANSMIT AT SAME TIME.
  - SWITCH CAN STORE FRAMES + ONLY FORWARD TO A PORT WHEN IT'S IDLE.
    ⇒ REALLY HARD TO HAVE A COLLISION
- SWITCH RECEIVES FRAME, LOOKS AT DEST ADDRESS, DECIDES WHICH PORT ON WHICH TO SEND FRAME

"MAC LEARNING:"
- SWITCHES "LEARN" WHICH HOST(S) LIVES ON WHICH PORT
  - IF DON'T KNOW, FLOOD TO ALL PORTS!
    ⇒ JUST AN OPTIMIZATION VS. OLD VERSION!

MAC LEARNING: NOW IT WORKS.



MAC ADDR	→ PORT
H_2	2
H_5	4
H_6	4

$H_2 \rightarrow H_3$ ON PORT 2

IF TABLE IS EMPTY
=> FLOOD

$H_5 \rightarrow H_2 \Rightarrow$ SEND TO PORT 2

MAC table example

```
R6#sh mac-address-table
EHWIC: 0
Destination Address Address Type VLAN Destination Port
------------------- ------------ ---- ----------------
5c45.27e0.8383            Dynamic         1 GigabitEthernet0/1/3
7641.7b63.584a            Dynamic        20 GigabitEthernet0/1/3
5c45.27e0.8381            Dynamic        10 GigabitEthernet0/1/3
0000.5e00.0101            Dynamic        10 GigabitEthernet0/0/1
ca3f.aee3.e3e6            Dynamic        20 GigabitEthernet0/1/3
644b.f012.7f75.           Dynamic        20 GigabitEthernet0/1/3
f018.9815.8eb8            Dynamic        20 GigabitEthernet0/1/3
ecb5.fa13.4677            Dynamic        20 GigabitEthernet0/0/2
a0a4.c5c2.4165            Dynamic        20 GigabitEthernet0/0/1
4c71.0c92.4f10            Dynamic        10 GigabitEthernet0/1/3
12d3.acae.bbc0            Dynamic        20 GigabitEthernet0/0/1
04d4.c448.9cf7            Dynamic        20 GigabitEthernet0/1/3
```

What if you WANT to talk to multiple hosts?

This is still possible! Separating collision domains means we don't need to do this unless necessary.

There are special Ethernet addresses to reach multiple hosts
- Broadcast address: send to all hosts
- Multicast: send to a certain group of hosts

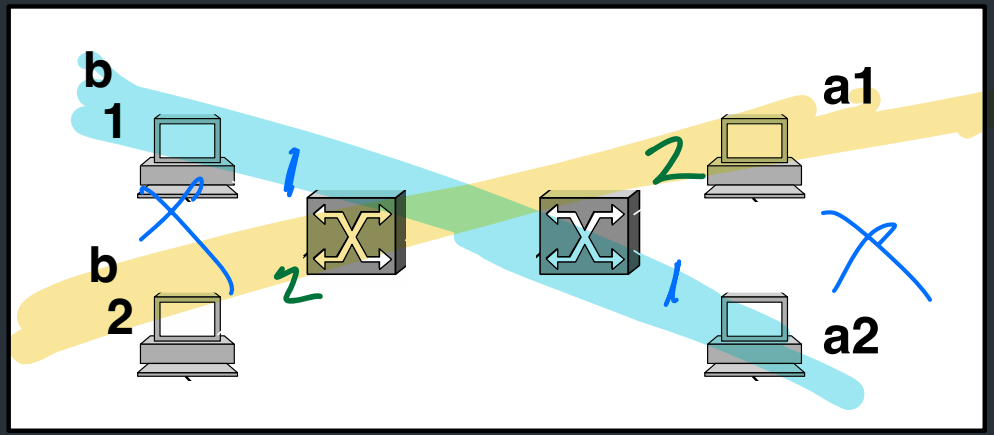
Attack on a Learning Switch

- Eve: wants to sniff all packets sent to Bob
- Same segment: easy (shared medium)
- Different segment on a learning bridge: hard
  - Once bridge learns Bob's port, stop broadcasting
- How can Eve force the bridge to keep broadcasting?
  - Flood the network with frames with spoofed src addr!

Also: VLANs

Consider: Company network, A and B departments

- Broadcast traffic does not scale
- May not want traffic between the two departments
- What if employees move between offices?

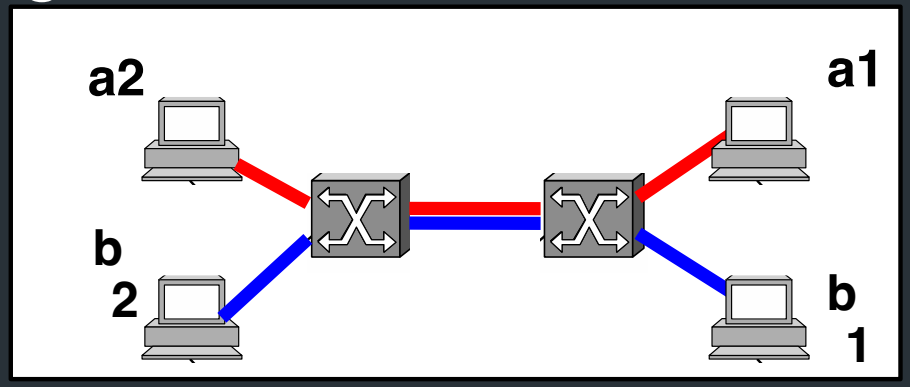


– DIFFERENT LOGICAL DOMAINS W/ SAME SWITCH/CABLING

VLANs

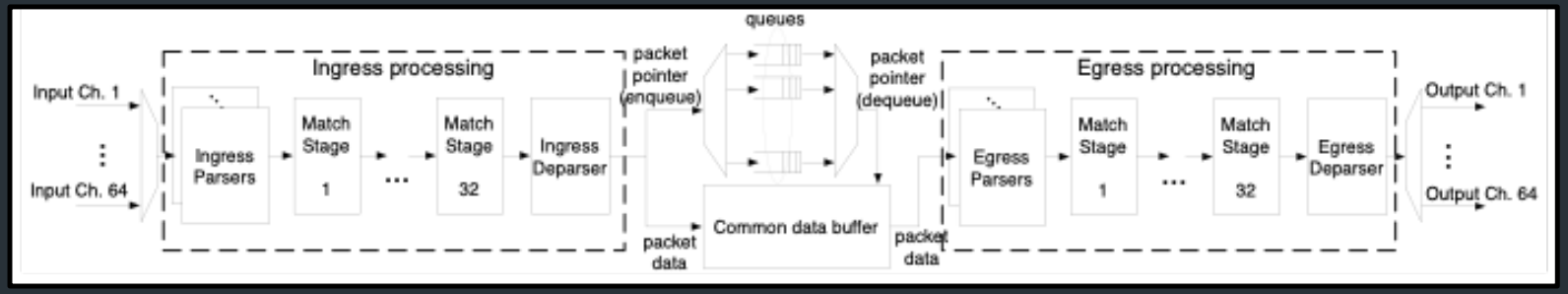
- Solution: Virtual LANs
  - Assign switch ports to a VLAN ID (color)
  - Isolate traffic: only same color
  - Some links may belong to multiple VLANs

=> Easy to change, no need to rewire



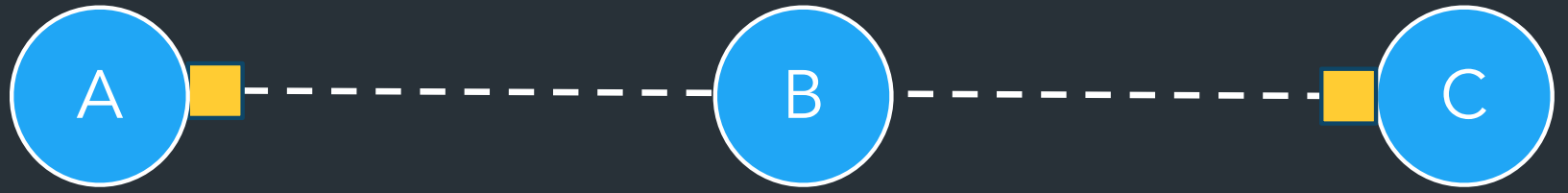
Current Developments

- Switches are becoming programmable
  - Match-action paradigm
  - Custom protocols, encapsulation, metering, monitoring

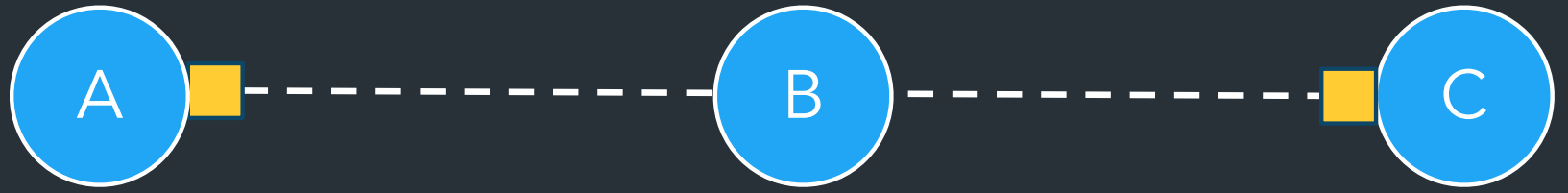


- Current speeds reach 12.8Tbps (32x400Gbps or 256x50Gbps) on a single programmable switching chip

How does this all change with wifi?



How does this all change with wifi?



Can’t detect collisions anymore!

=> Carrier Sense Multiple Access / Collision Avoidance

=> Try to send: if you don’t hear back, assume collision (and maybe retry)