CSCI-1680
Building Links and (Local) Networks

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

Administrivia

- Snowcast due Tuesday (2/10) by 11:59pm EST
- Look for announcement on Gradescope/testing soon
  - See our FAQ post for testing resources & common issues!

Next week: we start IP

Tuesday, 2/10

- Start of IP lectures
- Form for selecting your team
- Pre-release IP handout available

Next week: we start IP

Tuesday, 2/10

- Start of IP lectures
- Form for selecting your team
- Pre-release IP handout available

Thursday, 2/12

- Team form due
- IP project official release
- Gearup 5-7pm, CIT 165

Today

Last time: how to send over a link

Today: how to build *small* network?

- Link mechanics and sharing
- Case study/fundamental terms: Ethernet (and Wifi)
- How switching works

Fun systems facts: channels

Sending data takes time!

Latency:

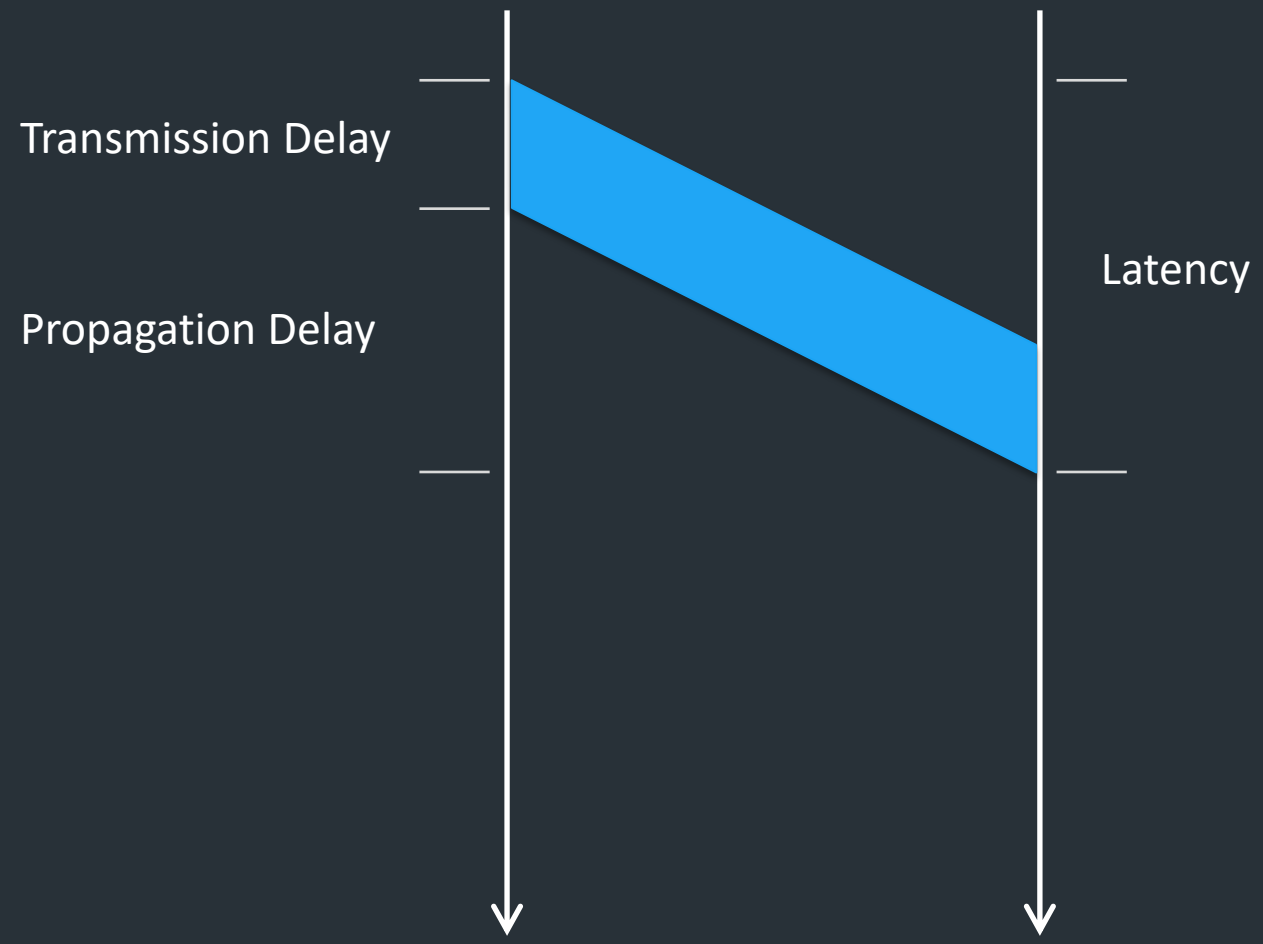
Sending data takes time!

Latency: time between sending data and when data arrives (somewhere)

Multiple components => many definitions, depending on what we're measuring

How to think about latency

How to think about latency



Latency: big picture

How to think about latency

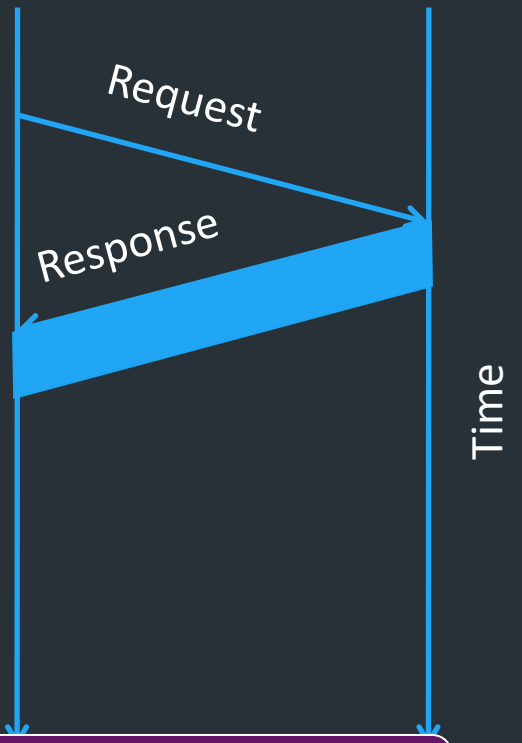
- Processing delay at the node: per message computation
- Queuing delay: time spent waiting in buffers
- Transmission delay: sending out the actual data
  - Size/Bandwidth
- Propagation delay: time for bits to actually go out on the wire
  - Upper bound?
  - Depends on media, ultimate upper bound is speed of light

Ping

Round trip time (RTT): time between request and response

Round trip time (RTT): time between request and response

When we design protocols,
can think about performance
based on number of RTTs

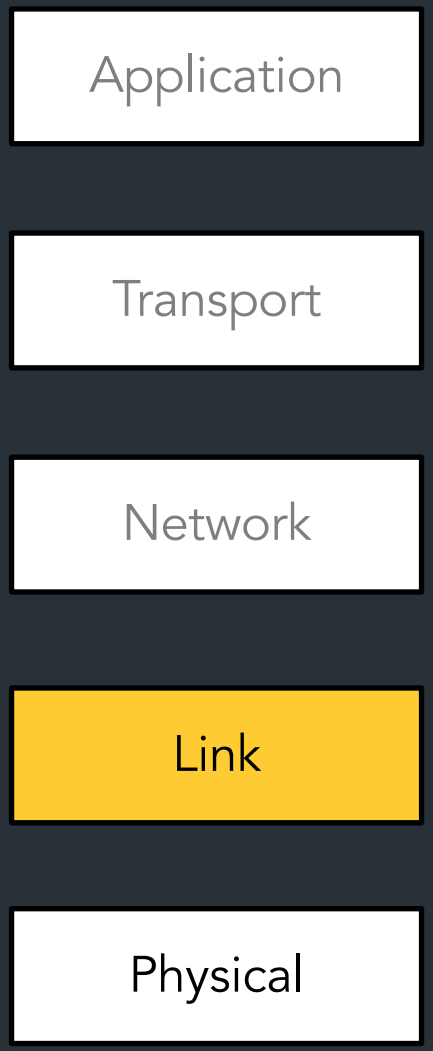


=> Not just about the physical layer!

RTT vs. Throughput

Link layer

What does “link layer” mean?



Service: user-facing application.
Application-defined messages

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

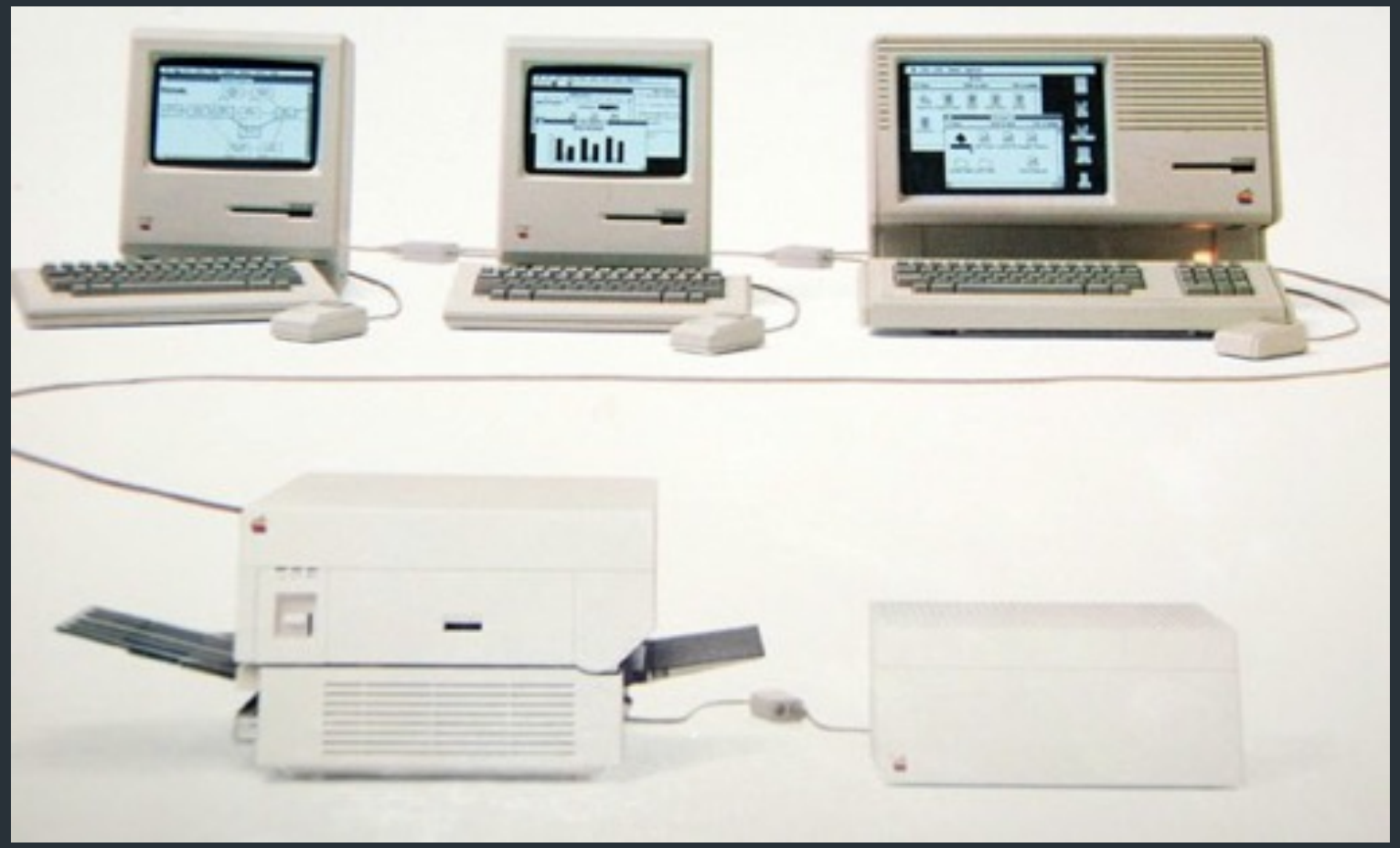
Service: move packets to any other node in the network
Internet Protocol (IP)

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

Service: move bits to other node across link

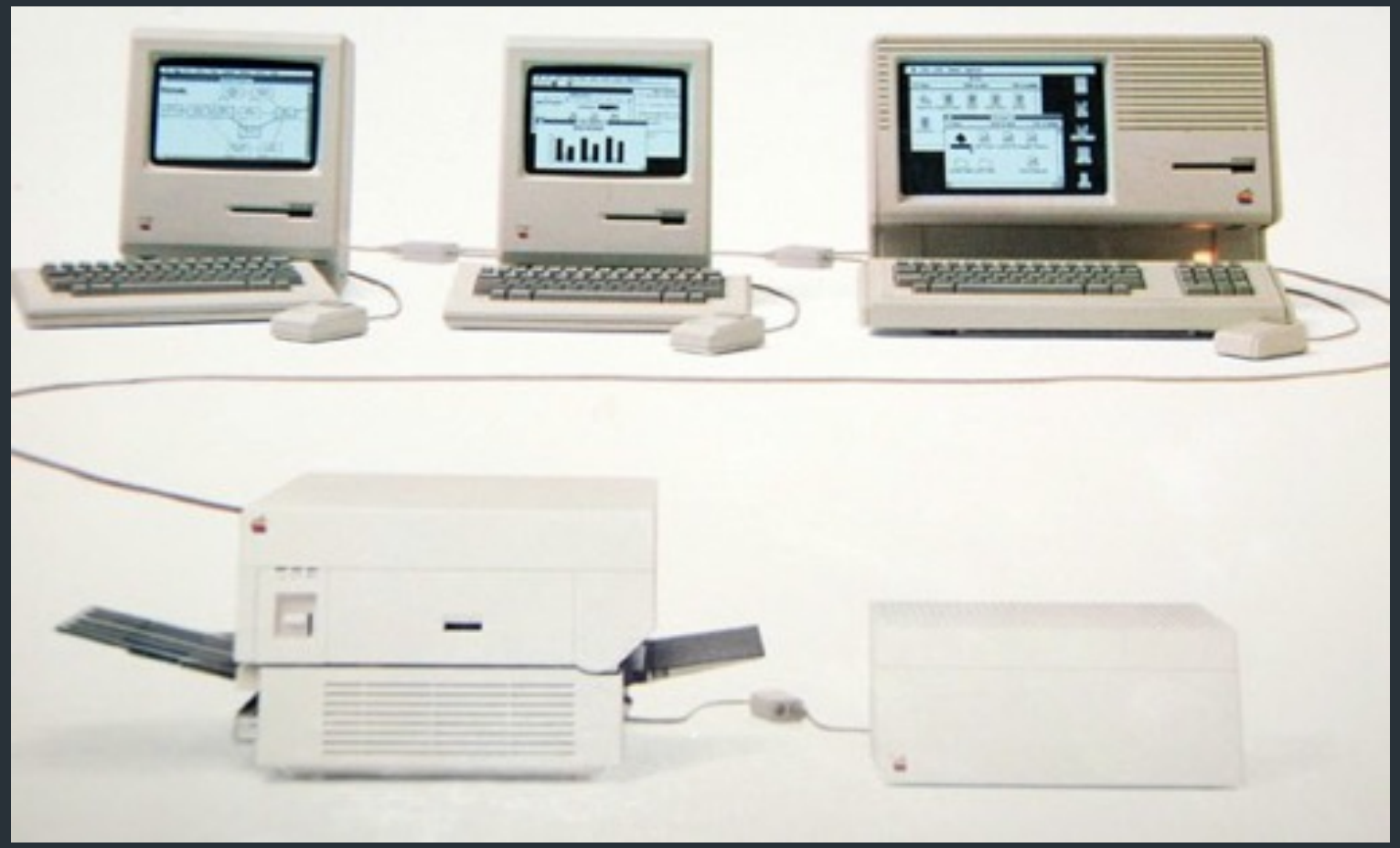
The main idea

Setting the scene



An AppleTalk network (1980s)

Setting the scene

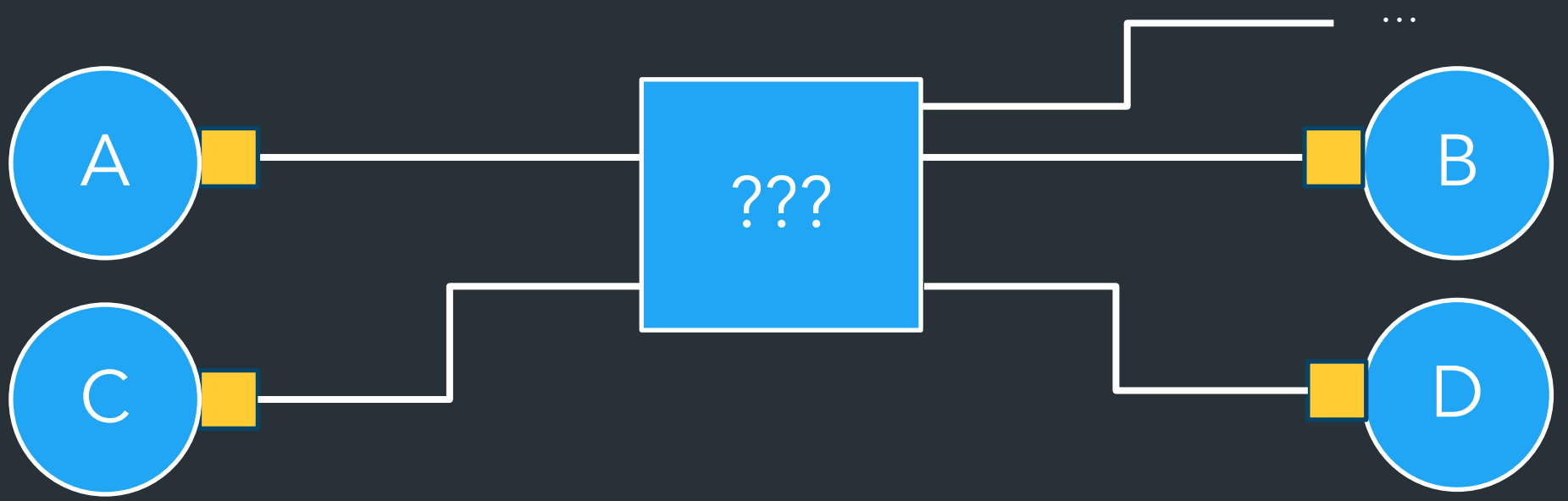


An AppleTalk network (1980s)

"Small" => Within a building, floor of office, etc

Related term: Local Area Network (LAN)

What does “link layer” mean?



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

How to share the channel?

How to share the channel?

Medium Access Control (MAC)

Medium Access Control

Medium Access Control

Idea: No more than one device can be “talking” at one time

Need a protocol for “who can talk when?”

Another 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)
- …

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
- …

Problems?

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

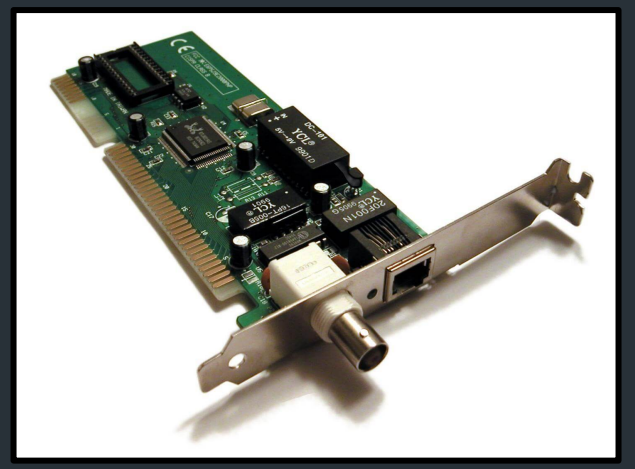
Why does this matter?

Different types of links solve these problems differently

- Ethernet (wired) vs. Wifi (wireless)
- Affects throughput, reliability, etc.

Understand why different links operate differently
=> How we build the Internet from them

How does a device use a link?

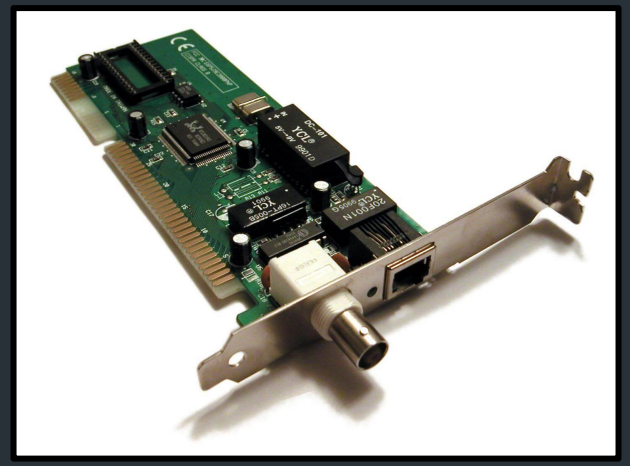



WLAN
Made in Malaysia
Dell P/N: 0PC193
MY-0PC19
MAC: 0018DE
REV:A02

Interface: device that connects something to a network

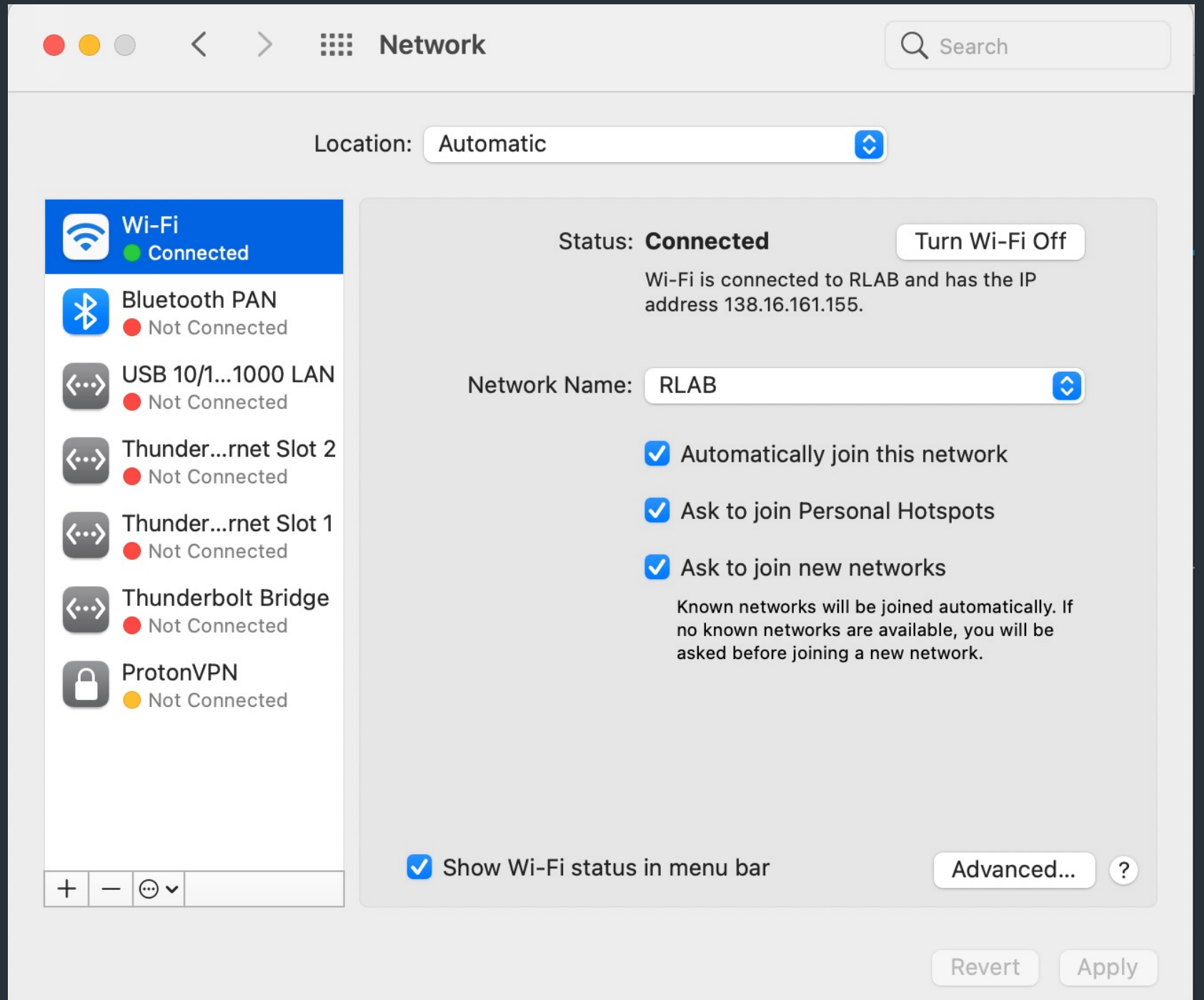
- OS abstraction for a network device
- Physical hardware that does the “talking”

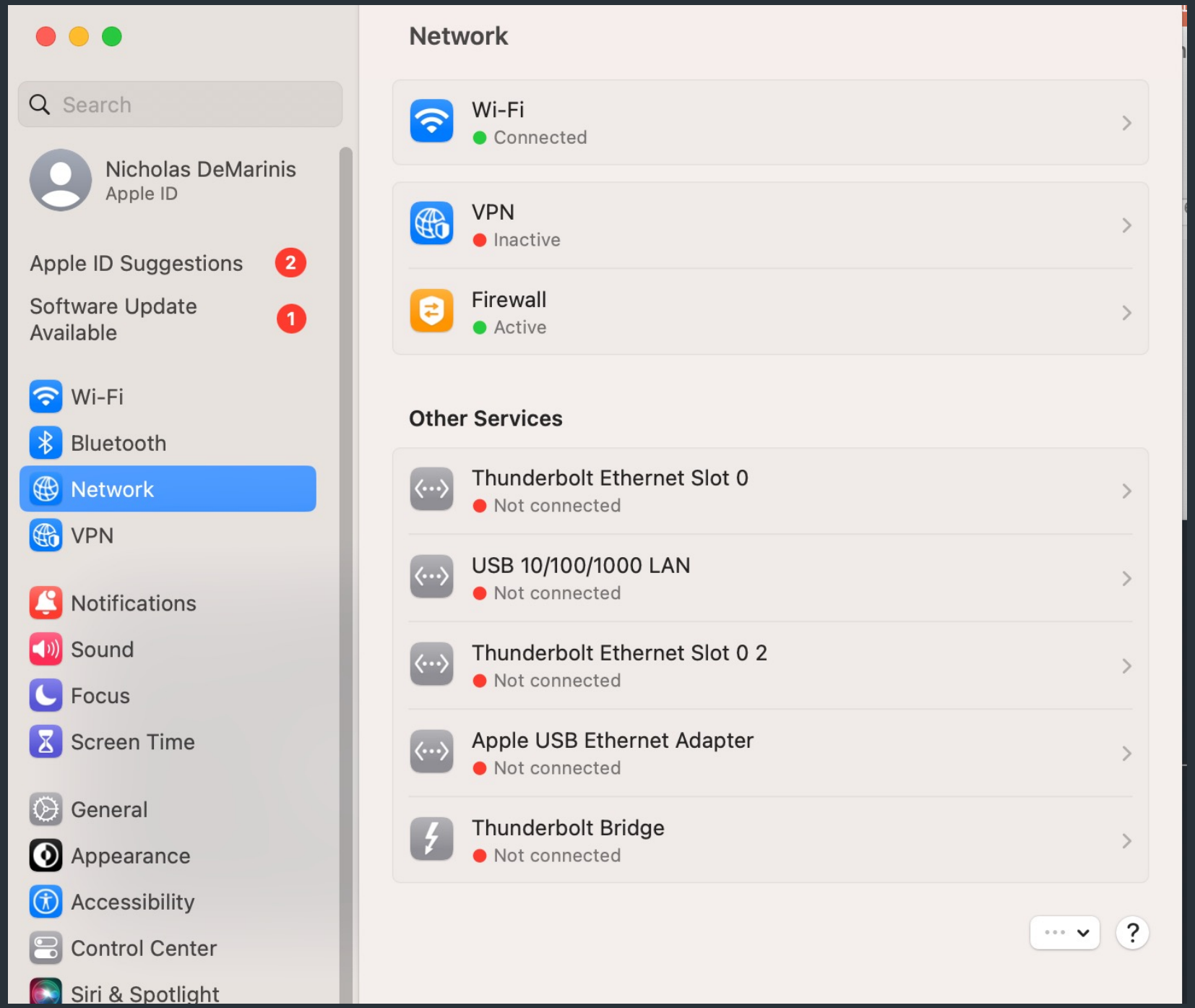
=> Network Interface Card (NIC)



Common interfaces

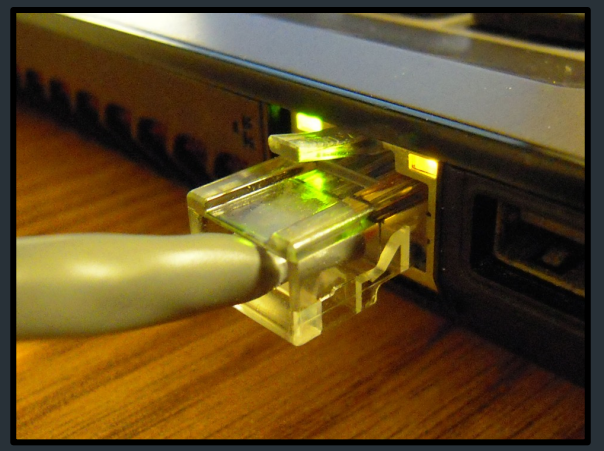
- Loopback: Virtual, only for local host
- Wifi, Ethernet, Bluetooth, …


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


Search
Nicholas DeMarinis
Apple ID
Apple ID Suggestions 2
Software Update Available 1
Wi-Fi
Bluetooth
Network
VPN
Notifications
Sound
Focus
Screen Time
General
Appearance
Accessibility
Control Center
Siri & Spotlight
Network
Wi-Fi
Connected
VPN
Inactive
Firewall
Active
Other Services
Thunderbolt Ethernet Slot 0
Not connected
USB 10/100/1000 LAN
Not connected
Thunderbolt Ethernet Slot 0 2
Not connected
Apple USB Ethernet Adapter
Not connected
Thunderbolt Bridge
Not connected

Example: Ethernet

Ethernet



Dominant wired LAN technology, has evolved significantly over time

- Original version (1983): 10Mbps
- Now (commonly): 1Gbps
- Also: 10Gbps, 40Gbps, …

New developments in physical media, encodings, hardware => higher speeds over time

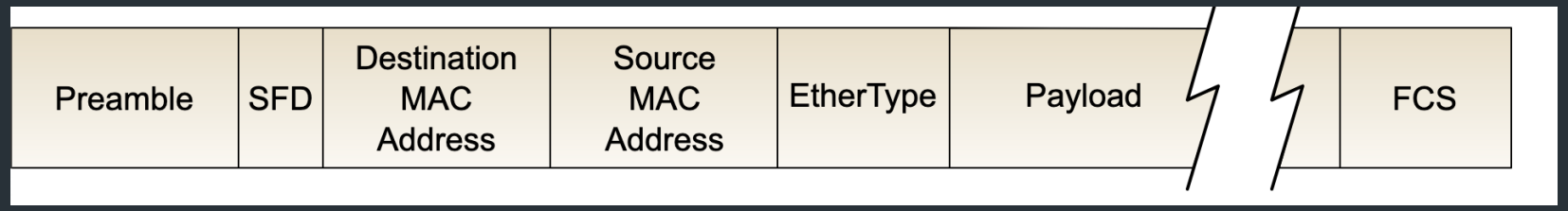
Ethernet: software viewpoint

- Logically all hosts are connected to each other

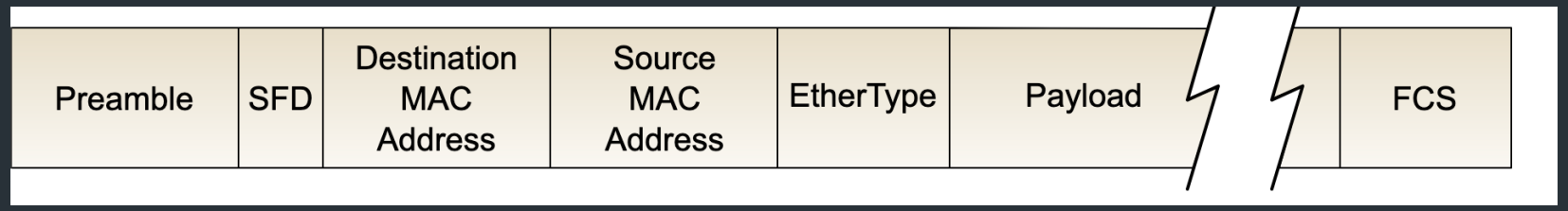
- All hosts have an “ethernet address” (“mac address”)
  => Globally-unique identifier

- If you know a host’s ethernet address, you can send to it

Ethernet: the header



Ethernet: the header

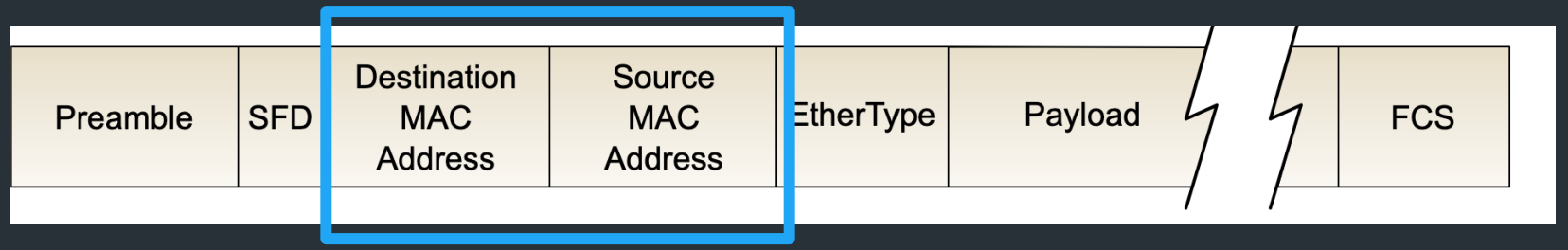


- Source address: where packet is from
- Destination address: where packet is going
  - ⇒ Devices ask: "Is this my packet?" "Where should I send this packet?"

<u>Other stuff</u>

- Preamble: when a packet starts
- FCS: Frame Check sequence (checksum)

Ethernet Addresses (mac addresses)

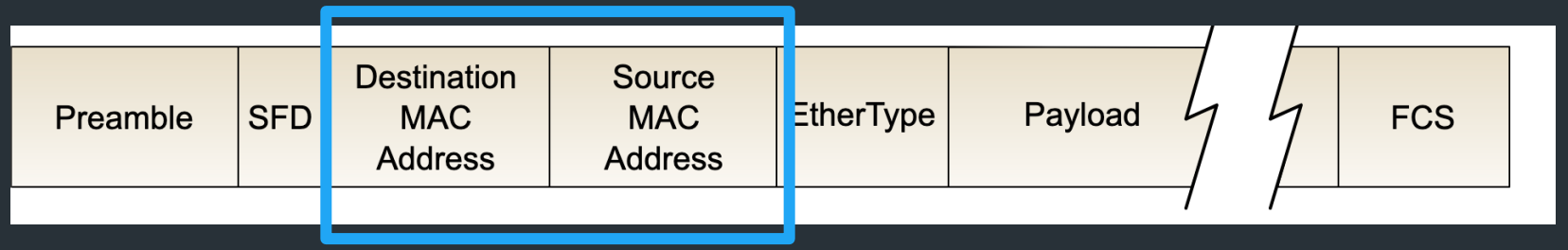


Globally unique, 48-bit address per interface

00:1c:43:00:3d:09 (Samsung adapter)

=> Nowadays, we call them “mac addresses” or “hardware addresses”

Ethernet Addresses (mac addresses)



Globally unique, 48-bit address per interface

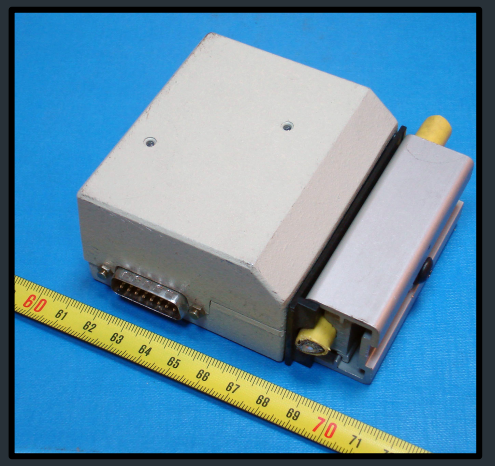
00:1c:43:00:3d:09 (Samsung adapter)

First 24 bits: Registered to manufacturers

=> Other protocols have adopted this address format (eg. Wifi, Bluetooth, …)

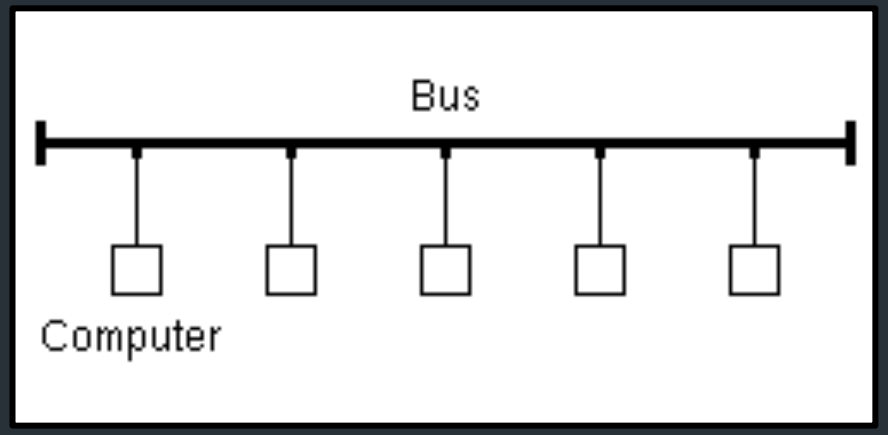
=> Nowadays, we call them “mac addresses” or “hardware addresses”

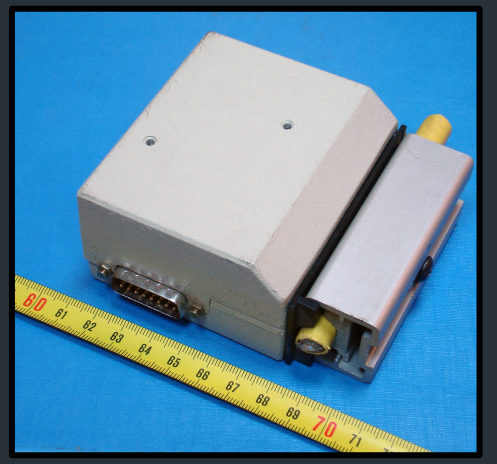
Ethernet's evolution



Ethernet's evolution

Originally, a shared medium with all hosts

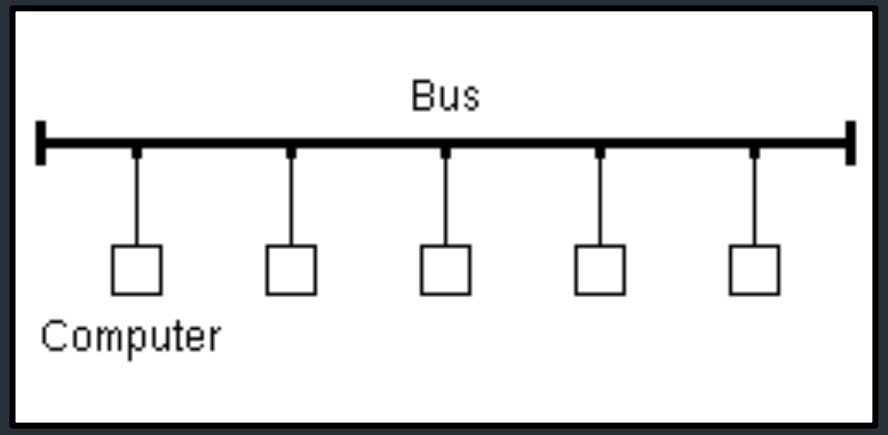


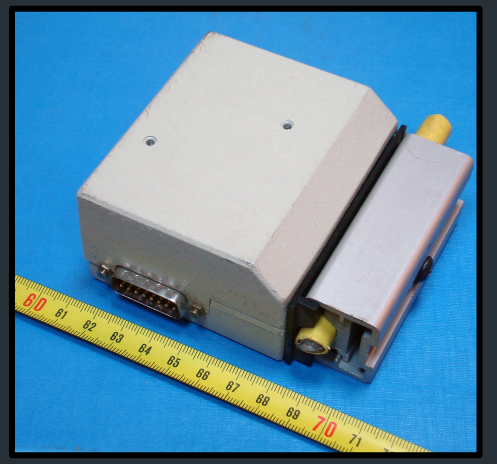


- Basic idea: all hosts can see all frames, read a frame if it matches your hardware address
- Implications?

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: all hosts in the same "collision domain"

Transmit algorithm

- If line is idle, transmit immediately
- Max message size: 1500 bytes
- If line is busy: wait until idle and transmit immediately

Classical Ethernet: Problems

Problem: all hosts in the same "collision domain"

Transmit algorithm

- If line is idle, transmit immediately
- Max message size: 1500 bytes
- If line is busy: wait until idle and transmit immediately

On Ethernet: generally possible to detect collisions (not always true!)

“Delay and try again later”

Sketch: In Ethernet

- *n*th 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)

“Delay and try again later”

Sketch: In Ethernet

- *n*th 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: 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”

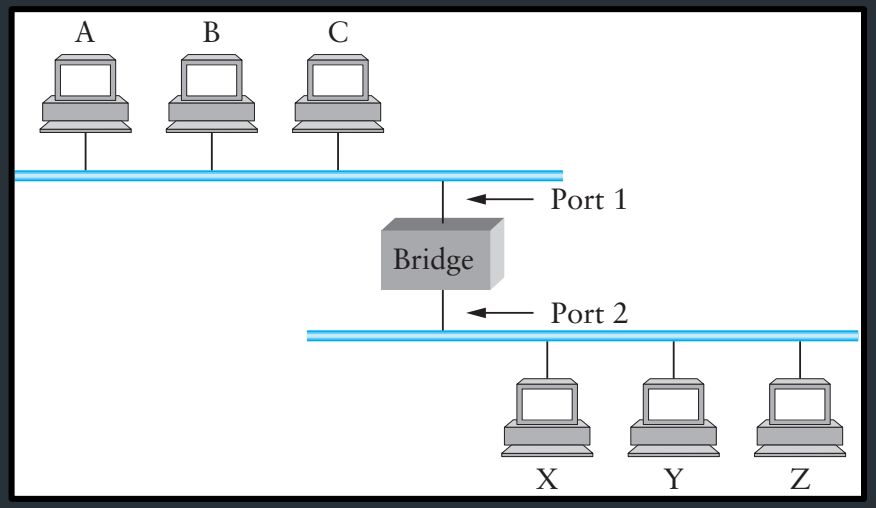
Avoiding collisions

- Early method: bridging

Avoiding collisions

Add some hardware to the network that can separate collision domains

Original way (1990s): bridges



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, how it works

MAC Learning

- Switches “learn” which host lives on which port by watching traffic

  =>Keeps table of <destination addr => port>

- 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!

MAC learning is just an optimization vs. old version

(but a pretty good one…)

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 can go wrong?

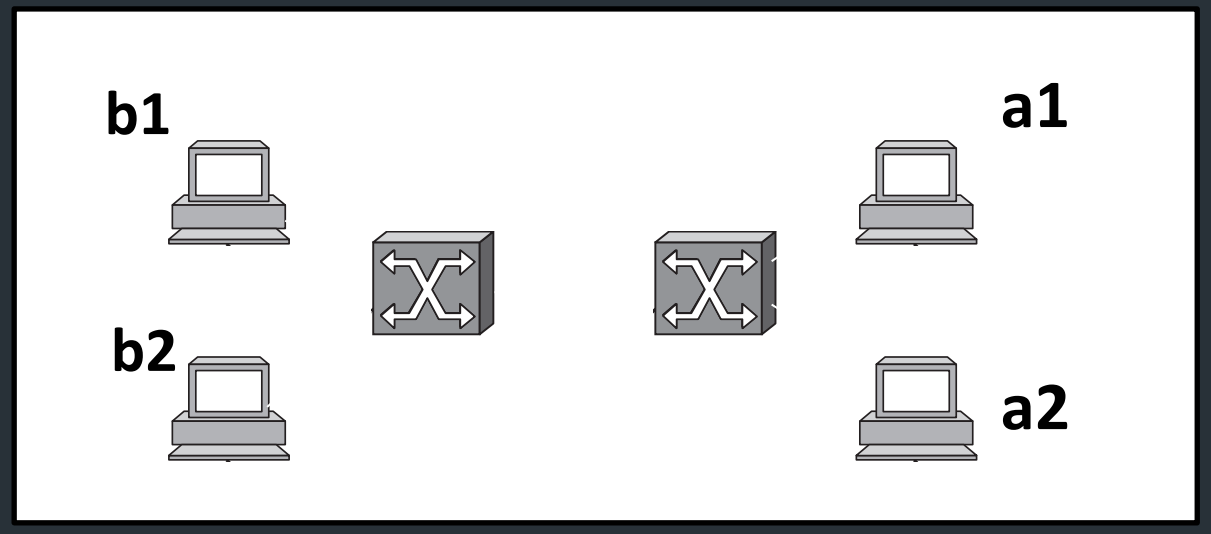
Attack on a Learning Switch

- 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

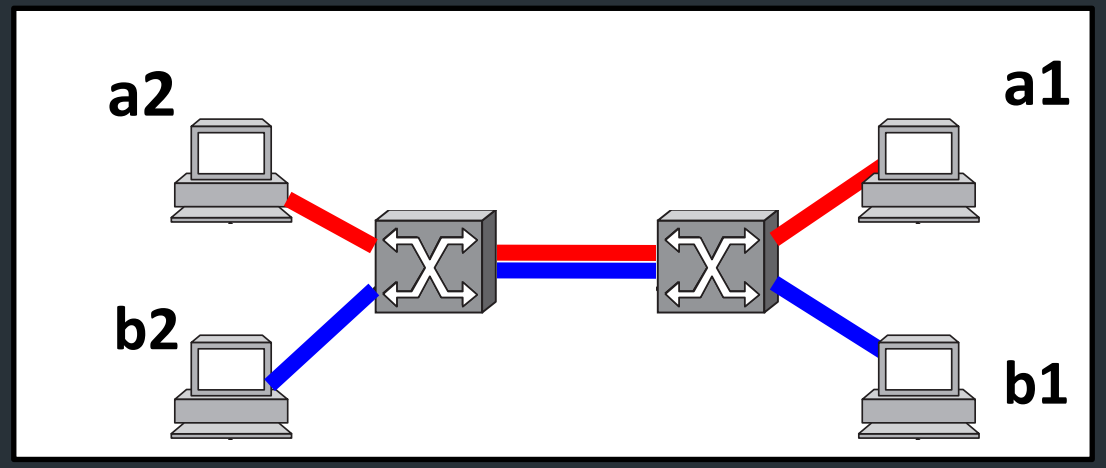
- Broadcast traffic does not scale
- May not *want* traffic between the two departments
- What if employees move between offices?



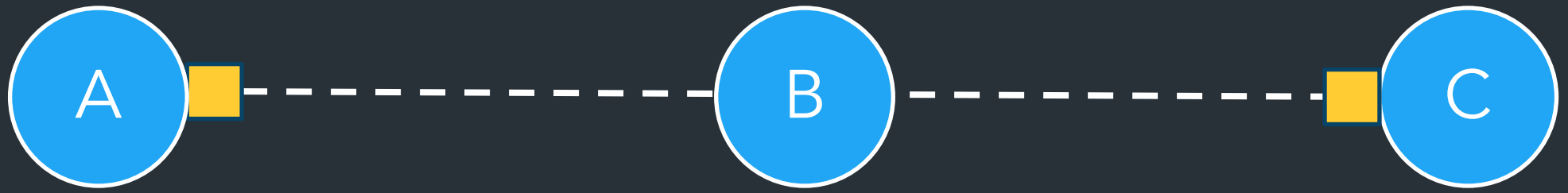
VLANs

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

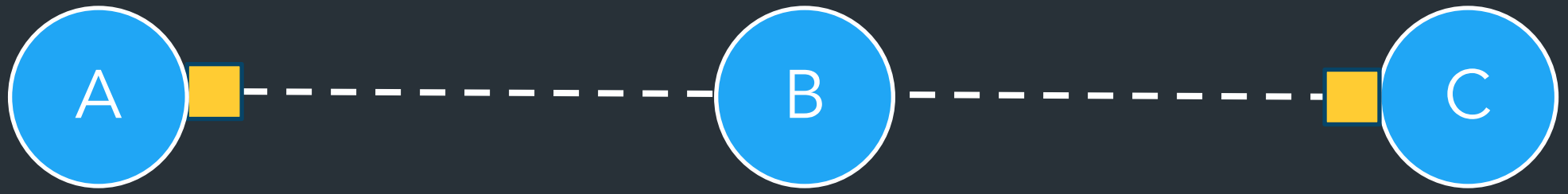
=> Easy to change, no need to rewire



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)

Extra material

Coming Up

- Connecting multiple networks: IP and the Network Layer