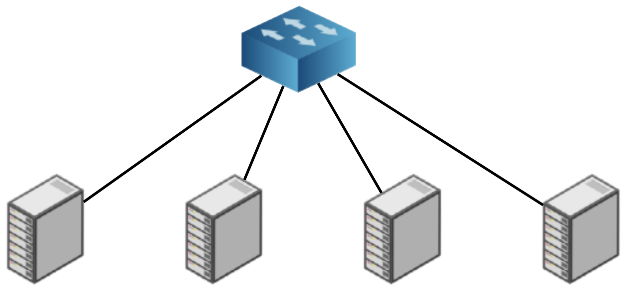
Computer Networks (WS23/24)

L4: The Link Layer - Part 2

Prof. Dr. Lin Wang
Computer Networks Group (PBNet)
Department of Computer Science
Paderborn University

Materials inspired by David J. Wetherall and Shyam Gollakota

Learning objectives



Application

Transport

Network

Data Link

Physical

Part 1
- Framing
- Error detection and correction
- Reliability: retransmission

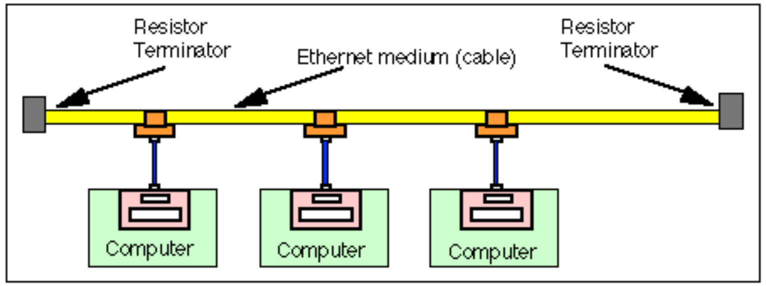
Part 2
- Multi-access
- Ethernet and switching
- Software-defined networking

Multi-access

Multiplexing

Sharing of a resource among multiple users

- Classic scenario is sharing a link among different users

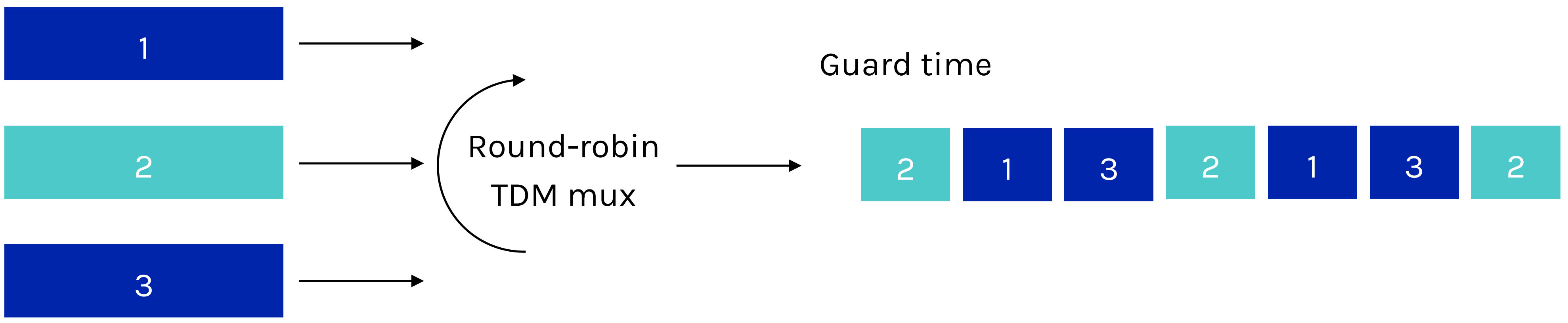


Static channel allocation

- Time Division Multiplexing (TDM)
- Frequency Division Multiplexing (FDM)

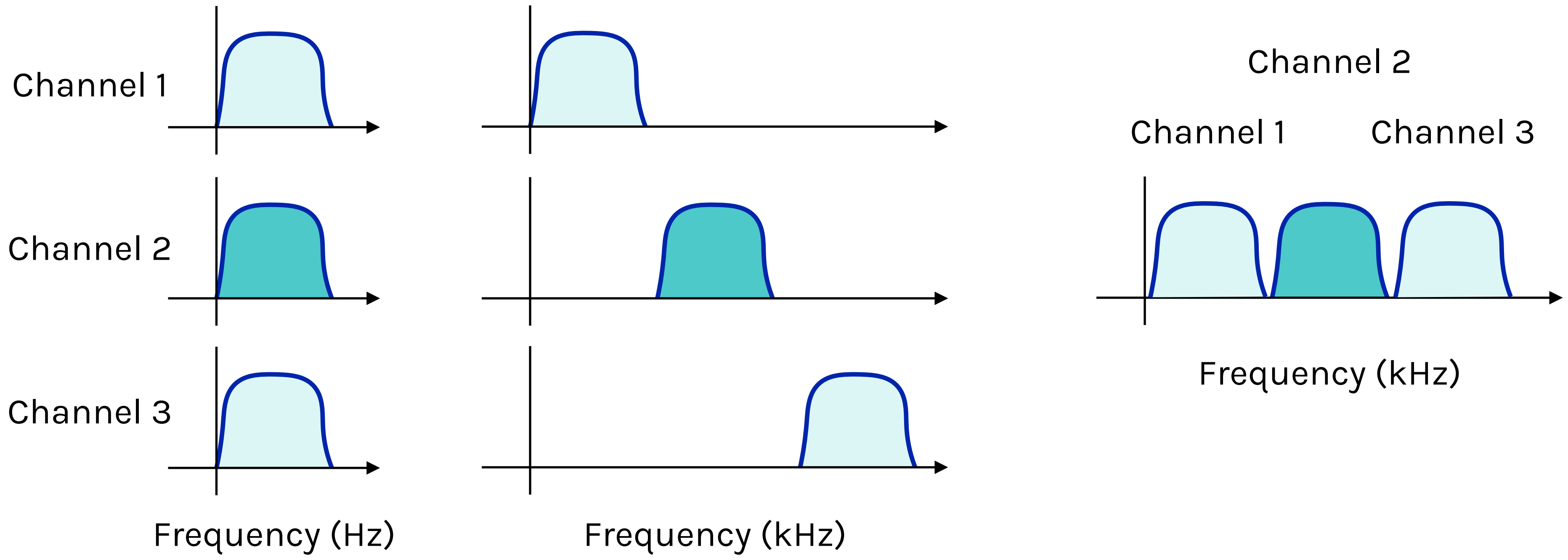
TDM

Users take turns on a fixed schedule

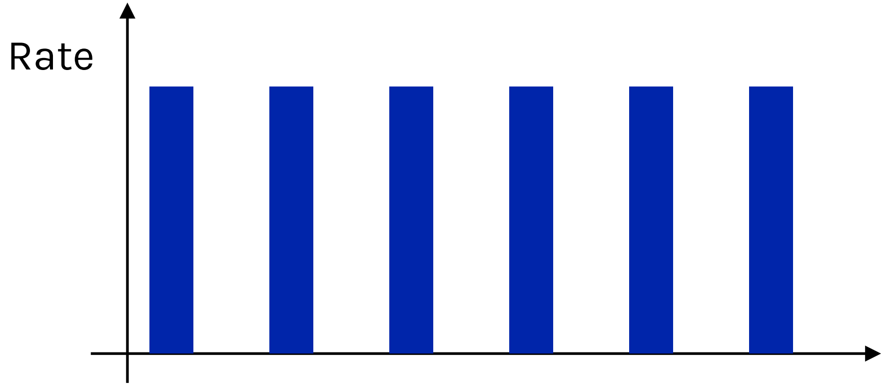


FDM

Put different users on different frequency bands



TDM vs. FDM



In TDM a user sends at a high rate for a fraction of time

In FDM a user sends at a low rate all the time

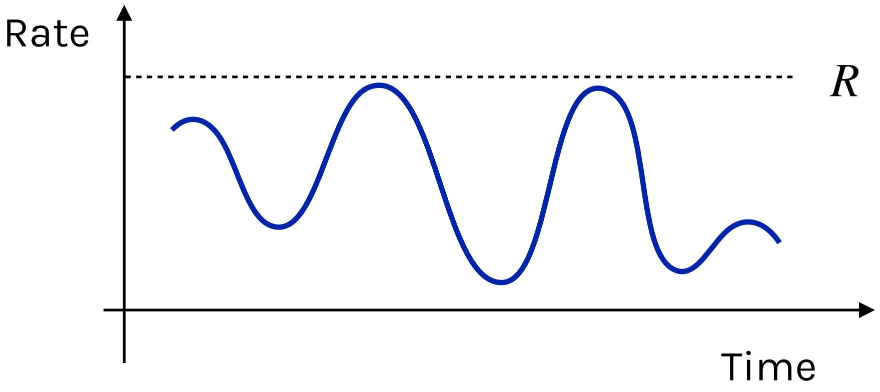
Statically divide a resource: suited for continuous traffic, fixed number of users

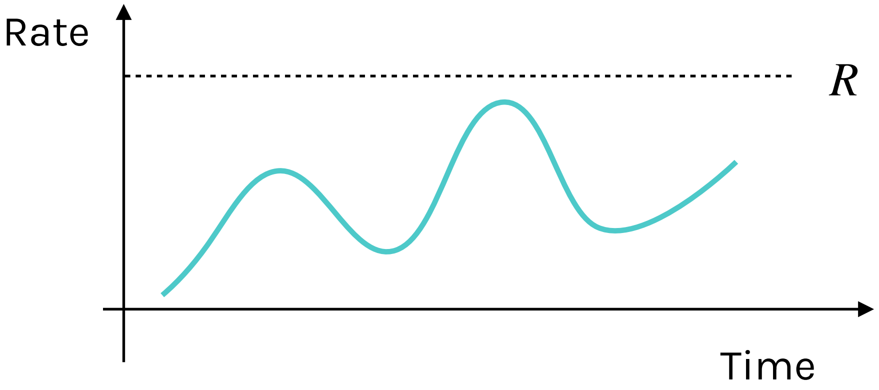
Widely used in telecommunications: TV and radio stations (FDM), GSM (TDM within FDM)

Multiplexing network traffic

Network traffic is bursty

- ON/OFF traffic sources: varying number of users
- Load varies greatly over time



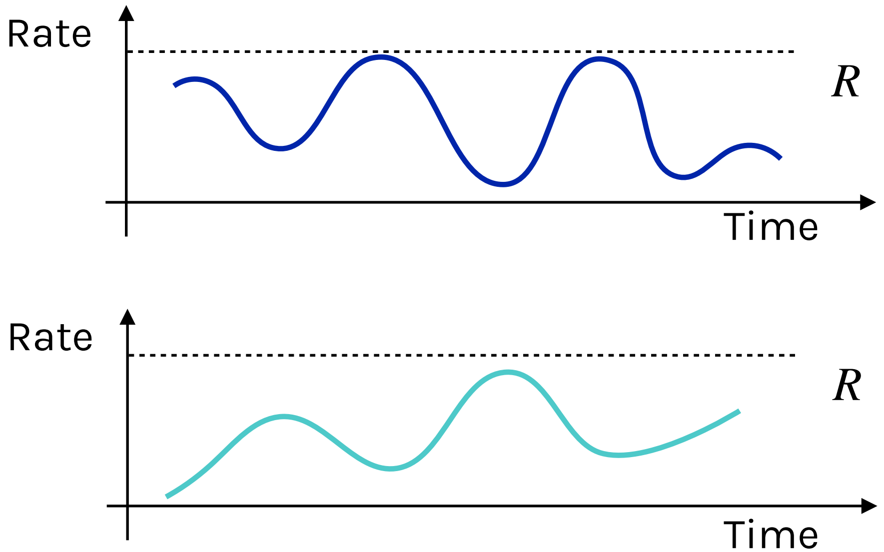


Inefficient to always allocate user their ON needs with TDM/FDM

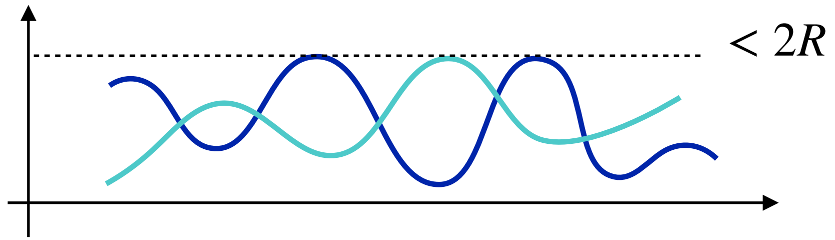
Multiplexing network traffic

Multi-access schemes multiplex users according to demands

- For gains of statistical multiplexing



Two users, each needs R



Together they need less than $2R$

Multi-access algorithms

Centralized: use a privileged "scheduler" to pick who gets to transmit and when

- Pros: scales well, usually efficient
- Cons: requirements management, fairness
- Examples: cellular networks (tower coordinates)

Distributed

- Pros: operates well under low load, easy to set up, equality
- Cons: scaling is hard!
- Example: WiFi networks

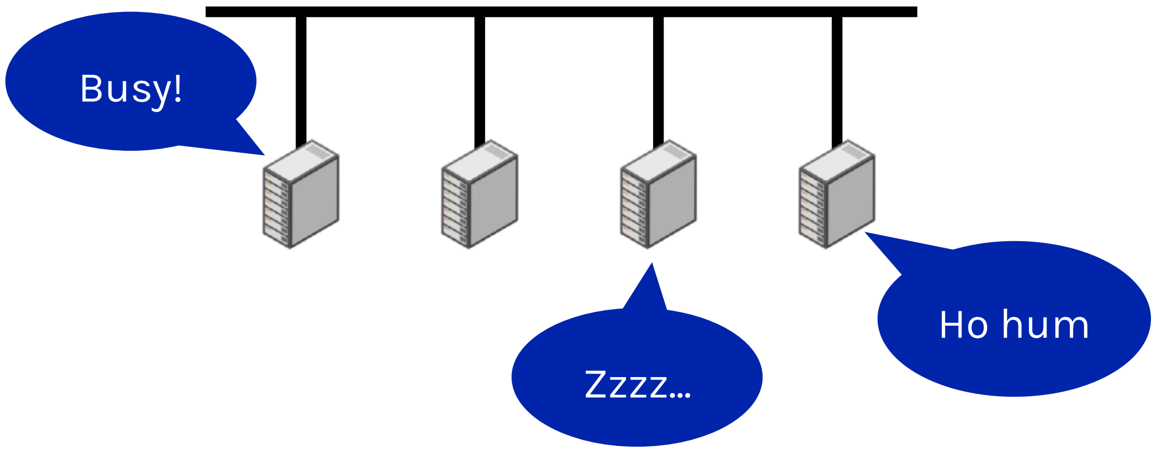
Distributed (random) access

How do nodes share a single link? Who sends when, e.g., in WiFi?

- Explore with a simple model
- Assume no one is in charge (i.e., distributed system)

Media-access control (MAC) protocols

- Basis for classic Ethernet
- Remember: data traffic is bursty



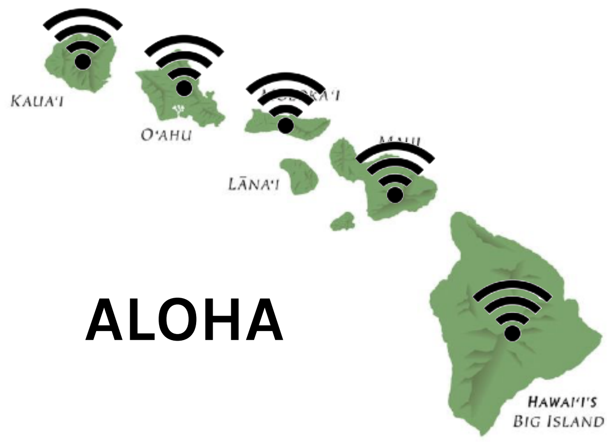
ALOHA network

Seminal computer network connecting the Hawaiian islands in the late 1960s

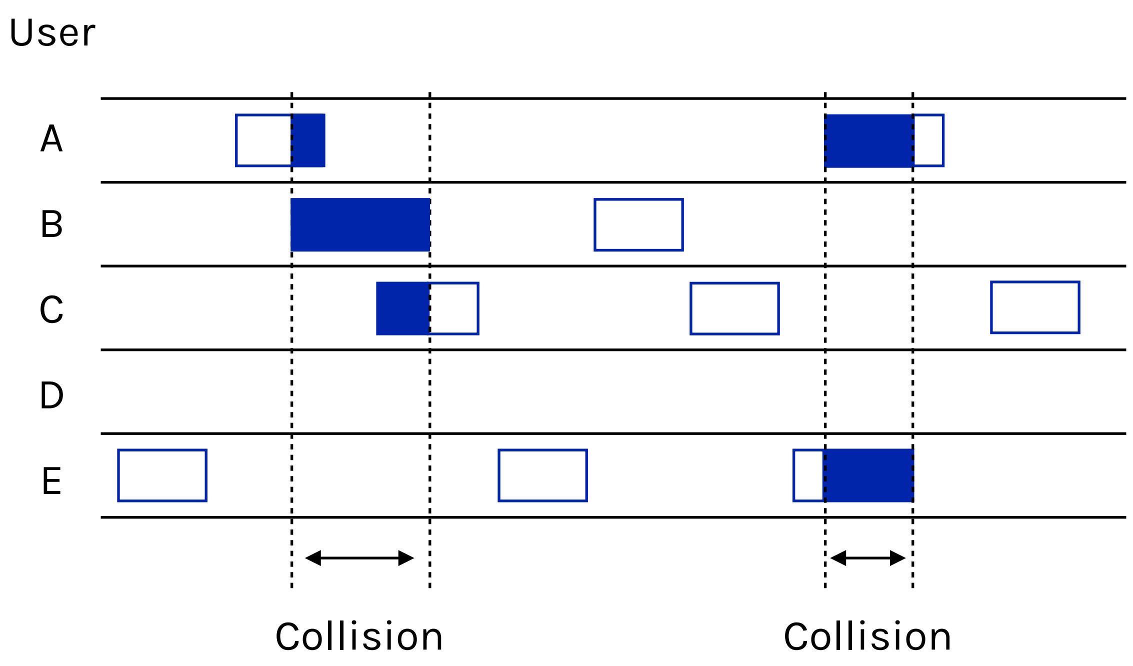
- When should nodes send?
- A new protocol was devised by Norm Abramson

Simple idea

- Node just sends when it has traffic
- If there was a collision (no ACK received) then wait a random time and resend



ALOHA protocol



Pros

- Simple, decentralized protocol that works well under low load

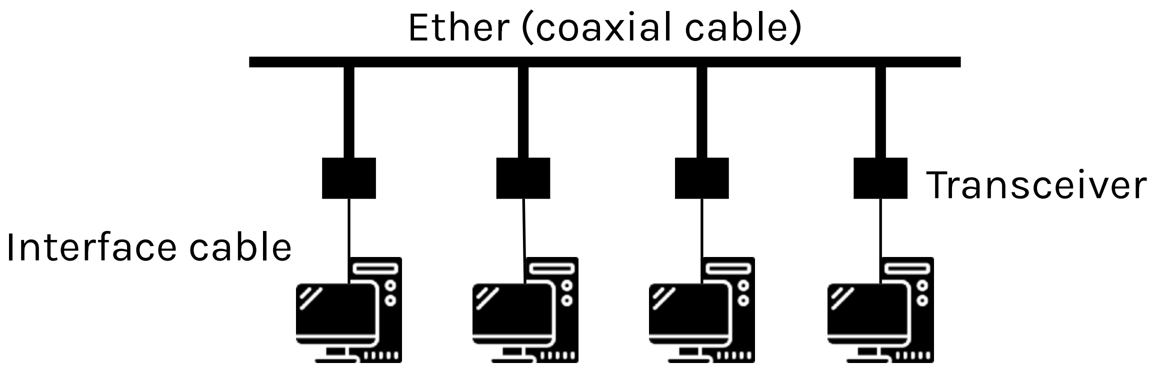
Cons

- Not efficient under high load (at most 18% efficiency)
- Divide time into slots and efficiency goes up to 36%

Classic Ethernet

ALOHA inspired Bob Metcalfe to invent Ethernet for LANs in 1973

- Nodes share 10 Mbps coaxial cable
- Hugely popular in 1980s, 1990s



Bob Metcalfe
(ACM Turing Award 2022)

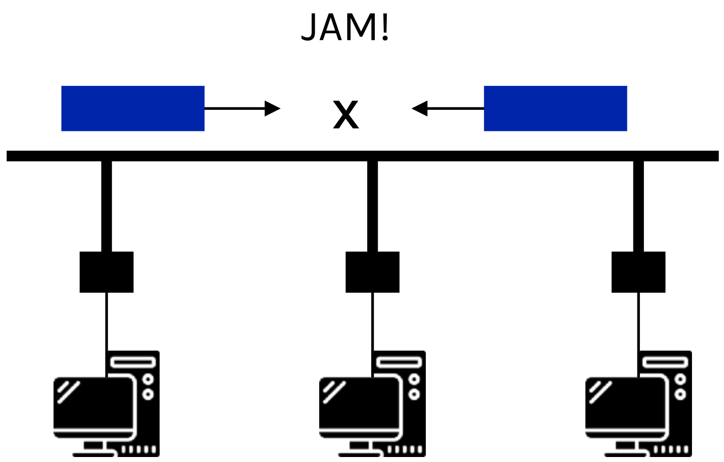
CSMA (carrier sense multiple access)

Improve ALOHA by listening for activity before we send!

- Can do easily with wires, not wireless

Does this eliminate collisions?

- Still possible to listen and hear nothing when another node is sending because of **delay**
- CSMA is a good defense against collisions only when BDP is small



CSMA "persistence"

What should a node do if another node is sending?

- Idea: wait until it is done, and send → 1-persistent CSMA

Problem: multiple waiting nodes will queue up then collide

- More load, more of a problem
- Intuition: if there are N queued senders, we want each to send next with probability $1/N$

p-persistent CSMA for slotted channels

- Upon idle channel, a node sends with a probability of p and defers sending until the next slot with a probability of $q = 1 - p$

CSMA "persistence"

Non-persistent CSMA

- A node sends when the channel is free
- If the channel is already in use, the node waits for a random period of time

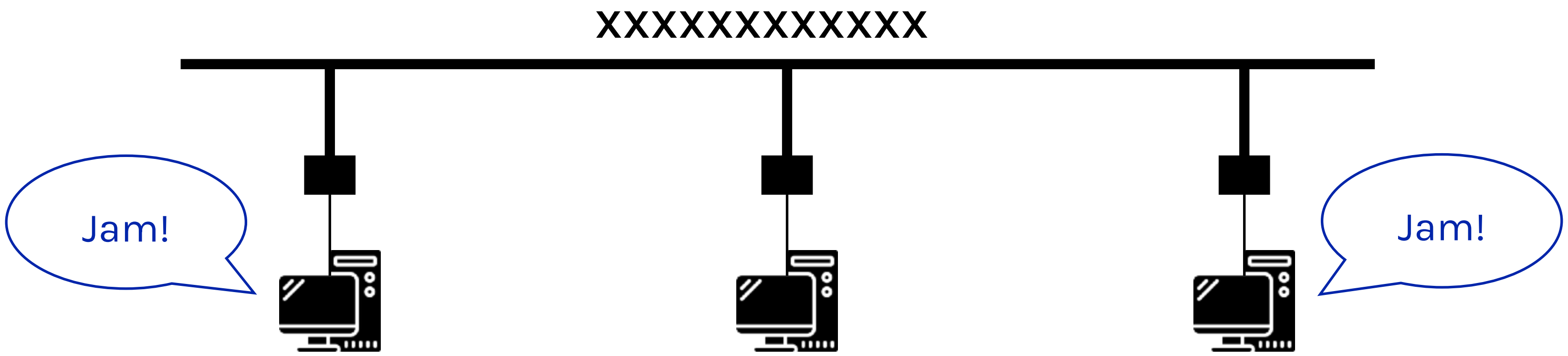
Binary Exponential Backoff (BEB)

- Doubles waiting interval for each successive collision
- Quicly gets large enough to work; very efficient in practice

CSMA/CD (with collision detection)

Can reduce the cost of collisions by detecting them and aborting (jam) the rest of the frame time

- Again, we can do this with wires



Everyone who collides needs to know it happened

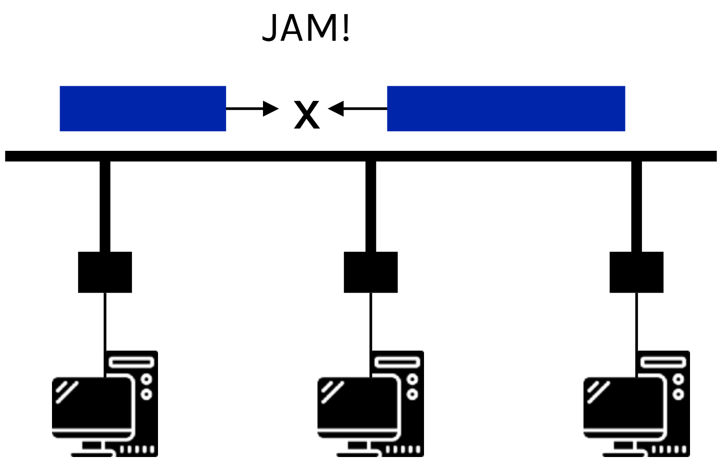
CSMA/CD complications

Everyone who collides needs to know it happened

- How long do we need to wait to know there was not a jam?
- Time window in which a node may hear of a collision (transmission + jam) is $2D$ seconds (D is the maximum delay in the network)

Impose a minimum frame length of $2D$ seconds

- So node cannot finish before detecting collision
- Ethernet minimum frame is 64 bytes
- Also sets maximum network length



Maximum network length

$$\text{Network length [m]} = \frac{\text{min\_frame\_size} * \text{speed of light}}{2 * \text{bandwidth}}$$

$$= 768 \text{ meters} \qquad \text{(for 100 Mbps)}$$

What about for 1 Gbps, 10 Gbps, and even 100 Gbps?

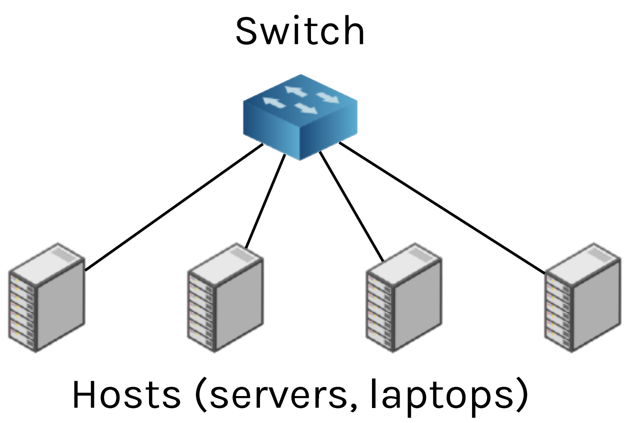
Ethernet (IEEE 802.3)

Classic Ethernet: most popular LAN of the 1980s, 1990s

- 10 Mbps over shared coaxial cable, with baseband signals
- Multi-access with "1-persistent CSMA/CD with BEB"
- CRC-32 for error detection, no ACKs or retransmission

Modern Ethernet

- Full-duplex
- Based on switches

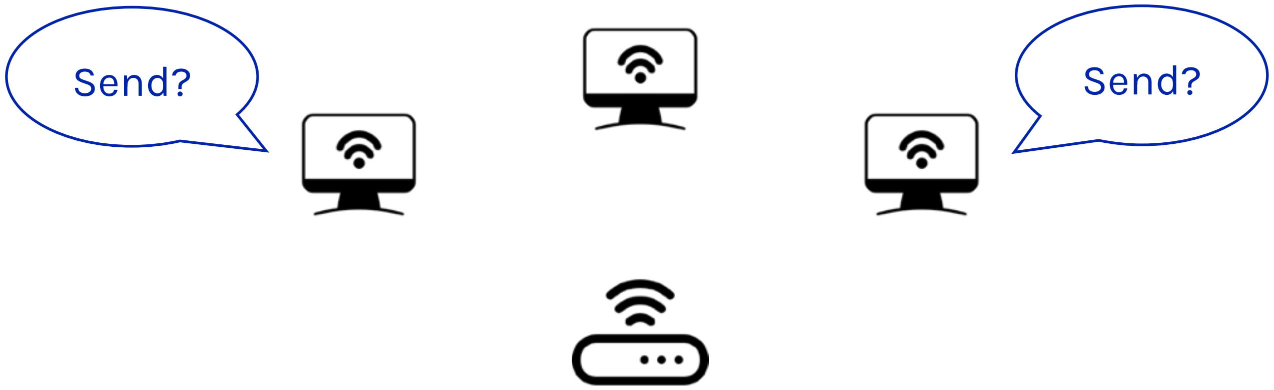


Do we still need multi-access protocols for modern Ethernet?

Multi-access in wireless networks

Wireless is more complicated than the wired case (surprise!)

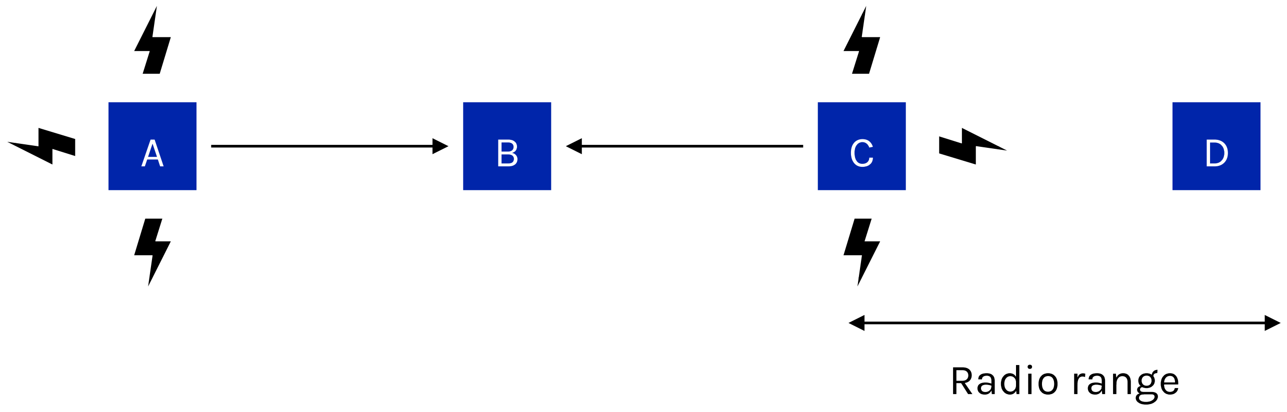
- Cannot carrier sense: do not know what's happening at the receiver
- Nodes cannot hear while sending: cannot collision detect



No carrier sense

Different coverage areas

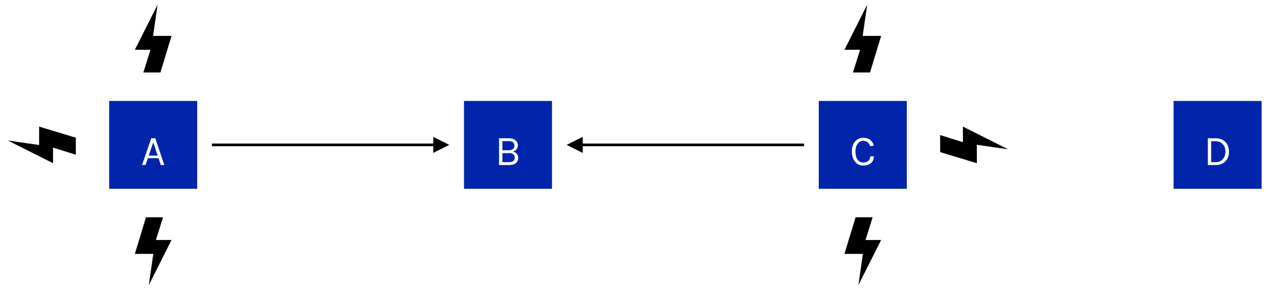
- Wireless signal is broadcast and received nearby, where there is sufficient SNR
- Hard to know potential competitors not within the radio range of itself



No carrier sense

Hidden terminals

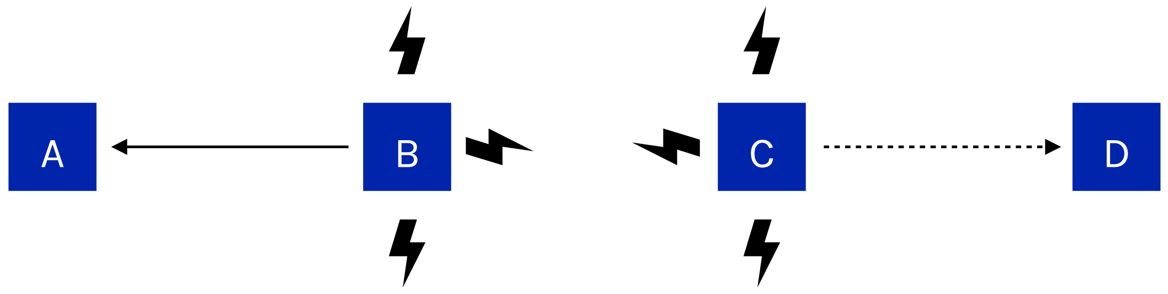
- Note A and C are hidden terminals when sending to B: cannot hear each other (to coordinate) yet collide at B
- We want to avoid the inefficiency of collisions



No carrier sense

Exposed terminals

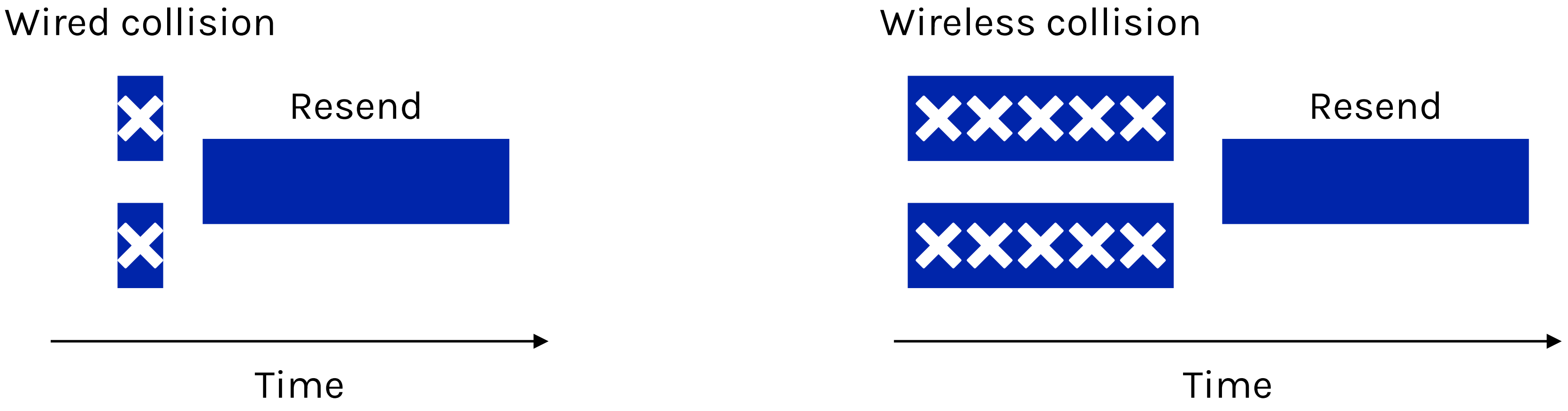
- B and C are exposed terminals when sending to A and D: can hear each other yet do not collide at receivers A and D
- We want to send concurrently to increase performance



Nodes cannot hear while sending

With wires, detecting collisions (and aborting) lowers their costs

More wasted time with wireless



MACA (multiple access with collision avoidance)

MACA uses a short handshake instead of CSMA (Karn, 1990)

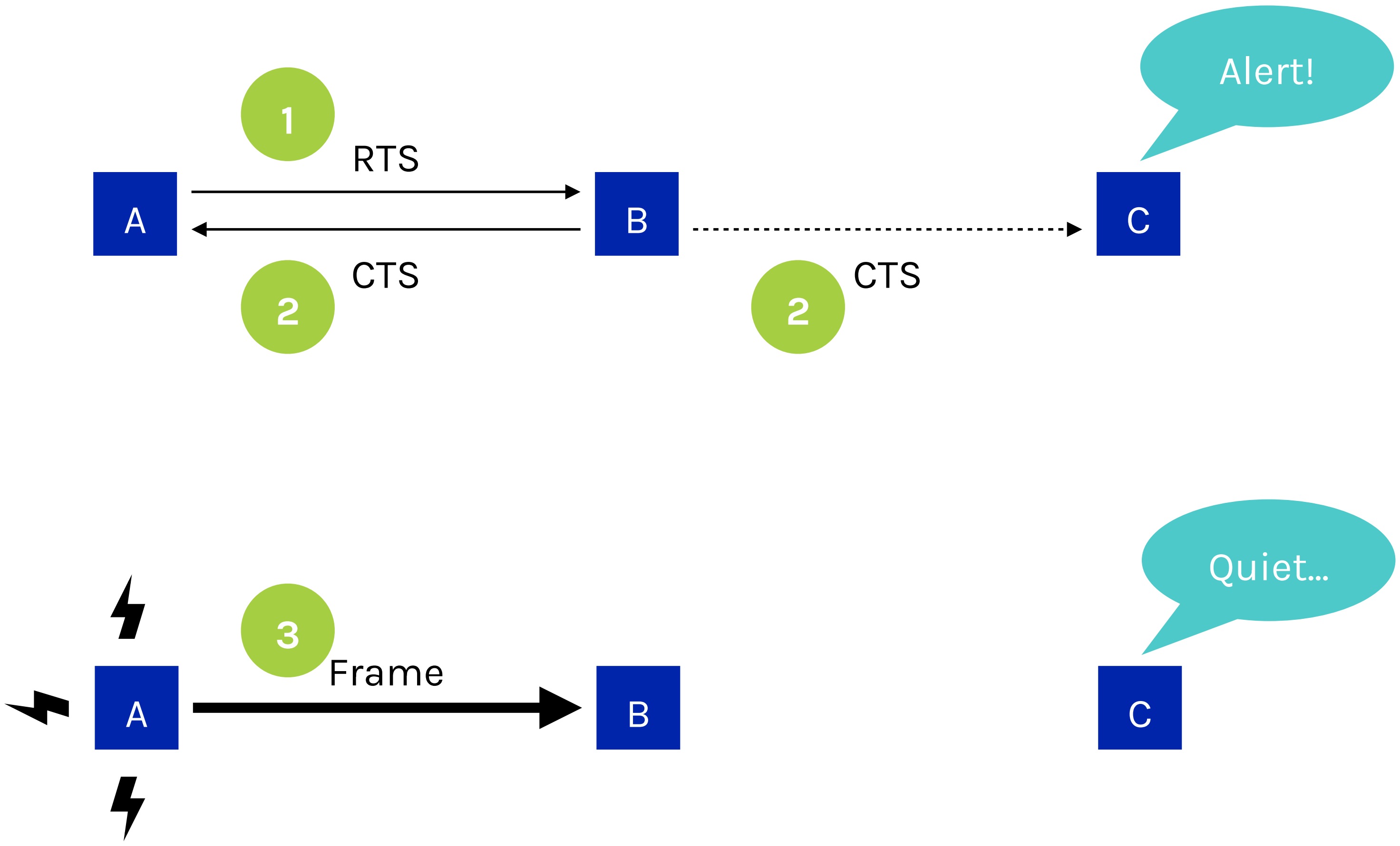
- 802.11 uses a refinement of MACA

Protocol rules

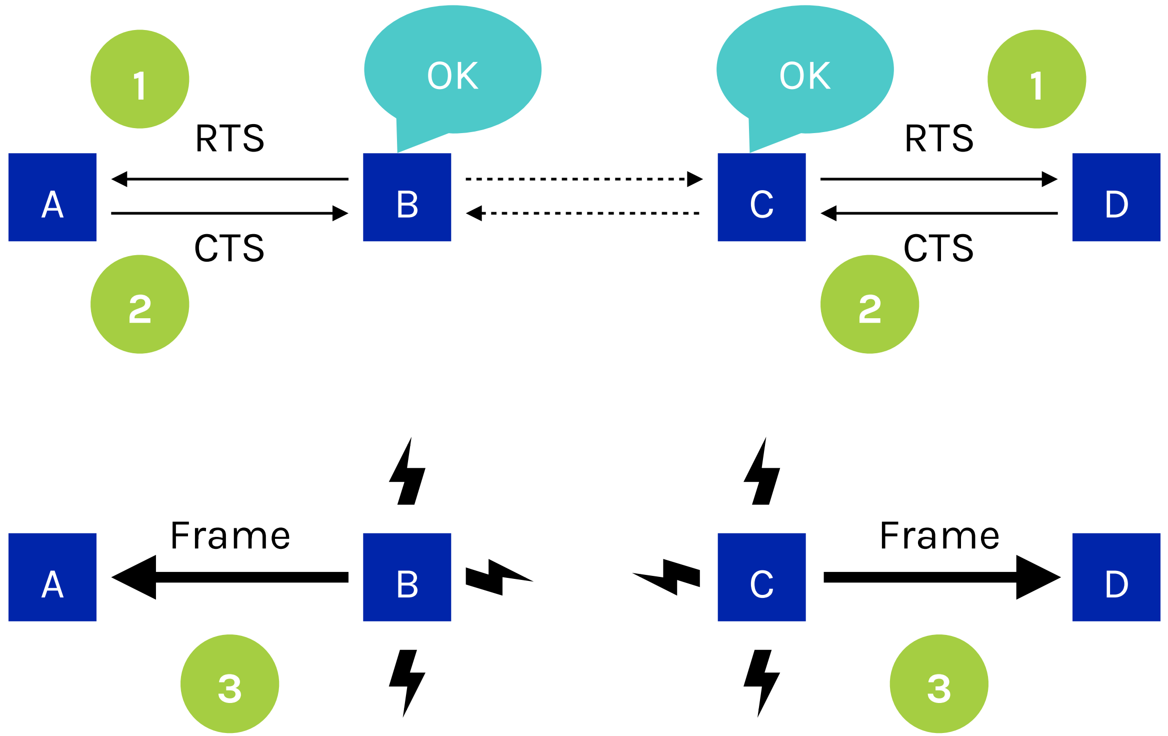
- A sender node transmits an RTS (Request-To-Send, with frame length)
- The receiver replies with a CTS (Clear-To-Send, with frame length)
- Sender transmits the frame while nodes hearing the CTS stay silent

Collisions on the RTS/CTS are still possible, but less likely

MACA: hidden terminals



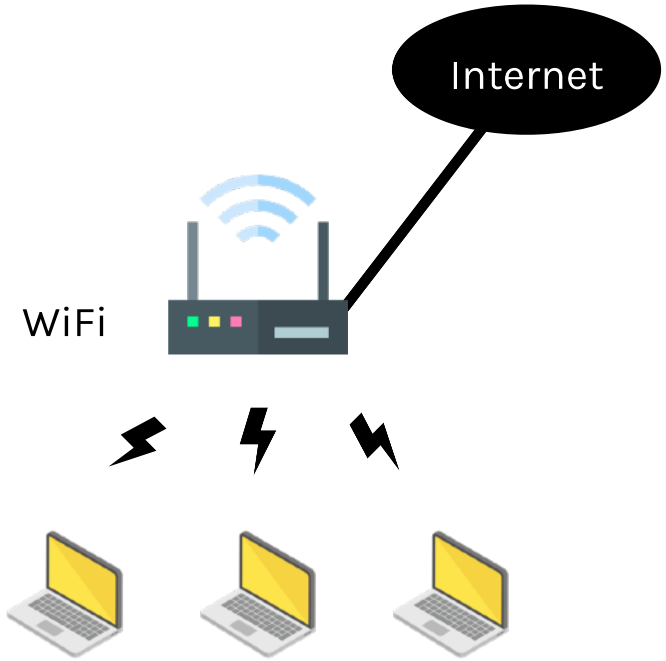
MACA: exposed terminal



802.11, or WiFi

Very popular wireless LAN started in the 1990s

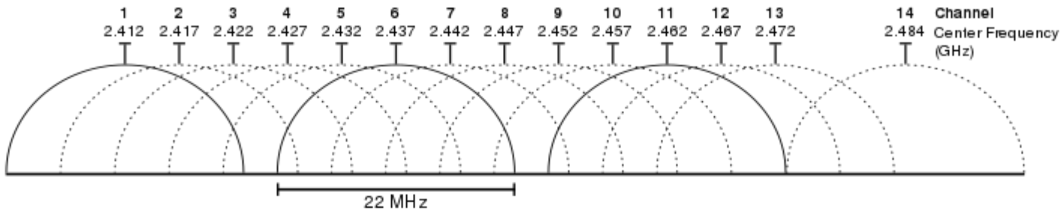
- Clients get connectivity from a (wired) access point (AP)
- A multi-access problem
- Various flavors have been developed over time: faster, more features



802.11 physical layer

Uses 20/40 MHz channels on ISM (unlicensed) bands

- 802.11b/g/n on 2.4 GHz
- 802.11 a/n on 5 GHz



OFDM modulation (except legacy 802.11b)

- Different amplitudes/phases for varying SNRs
- Rates from 6 to 54 Mbps, plus error correction
- 802.11n uses multiple antennas: lots of fun tricks here

802.11 link layer

Design choices

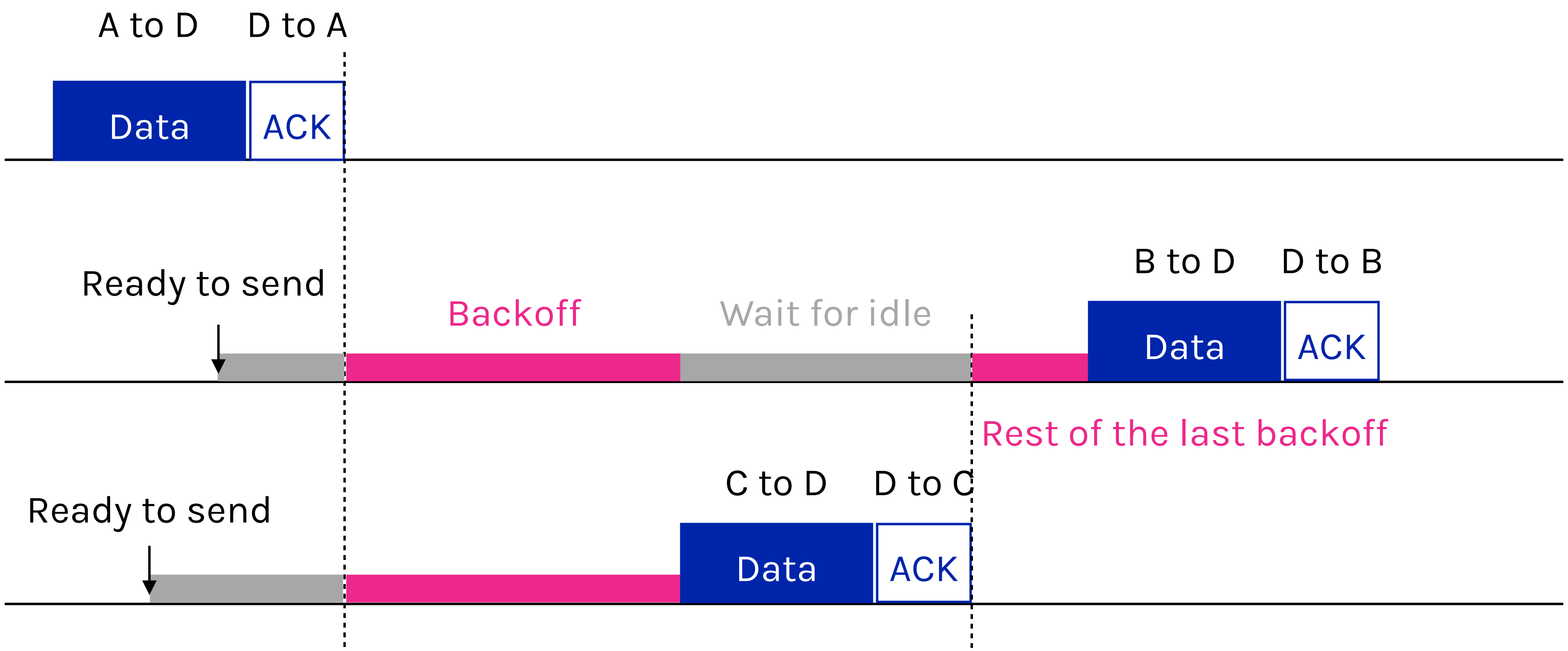
- Multi-access uses CSMA/CA; RTS/CTS optional
- Frames are ACKed and retransmitted with ARQ (why?)
- Funky addressing (three addresses!) due to AP
- Errors are detected with a 32-bit CRC
- Many other features, e.g., encryption, power saving

2	2	6	6	6	0 or 2	6	0 or 2	0 or 4	0 - 2312	4
Frame Control	Duration /ID	Address 1	Address 2	Address 3	Sequence Control	Address 4	QoS Control	HT Control	Frame Body	FCS

https://en.wikipedia.org/wiki/802.11_Frame_Types

802.11 CSMA/CA for multiple access

Still using Binary Exponential Backoff (BEB)!



Centralized MAC: cellular networks

Specturm suddenly very very scarse

- We cannot waste all of it sending JAMs

We have quality of service (QoS) requirements

- Cannot be as loose with expectations
- Cannot have traffic fail

We also have client/server

- Centralized control
- Not peer-to-peer, decentralized

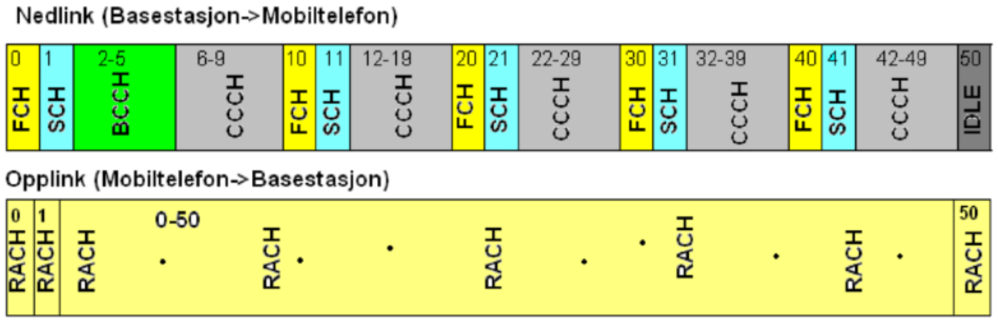
GSM MAC

Based on FDMA/TDMA

Use one channel for coordination - random access with BEB (no CSMA, cannot detect)

User other channels for traffic

- Dedicated channel for QoS

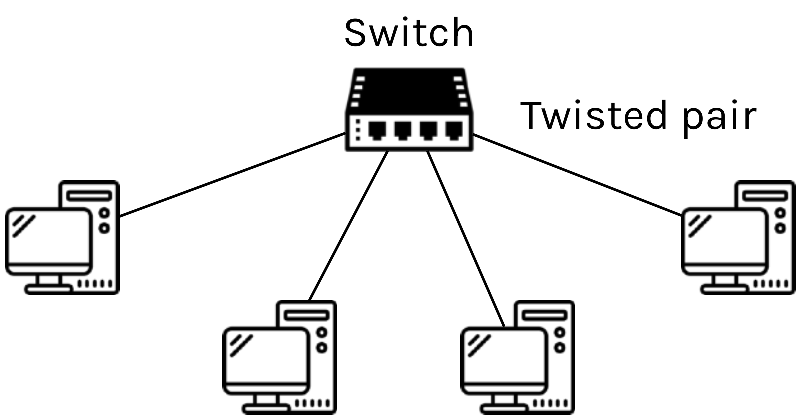


Ethernet and Switching

How NOT to multiple access?

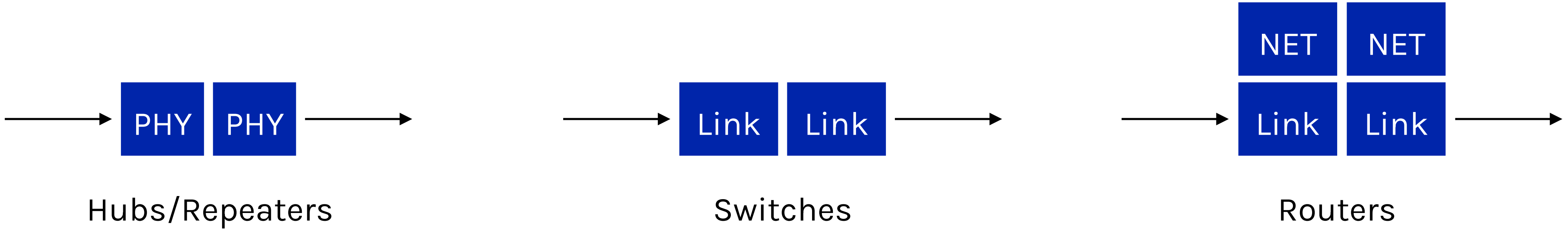
We use a special device called switch to interconnect end-devices

- Transmission are serialized on the link: **no collision** on the link any more
- Switches can interconnect multiple end-devices
- Basis of modern (switched) Ethernet

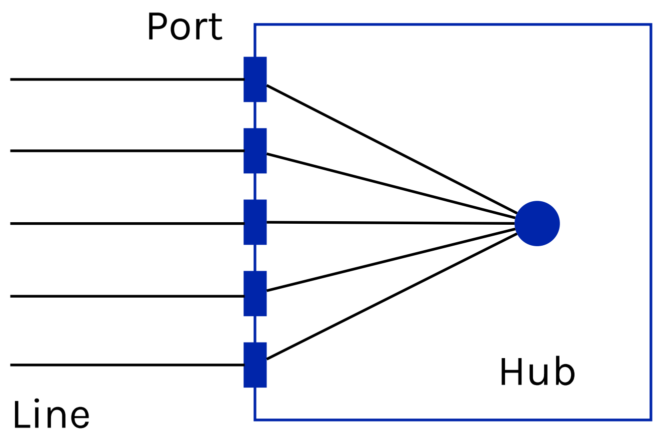


What's inside the box?

Hubs/Repeaters, switches, and routers all look similar, but they work on different layers

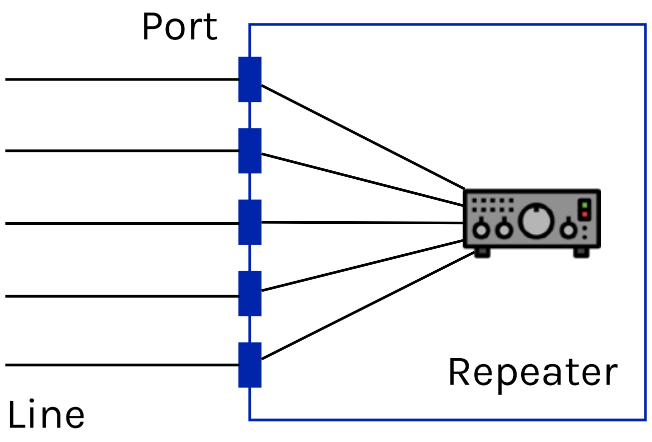


Inside a hub



All ports are wired together; more convenient and reliable than a single shared wire

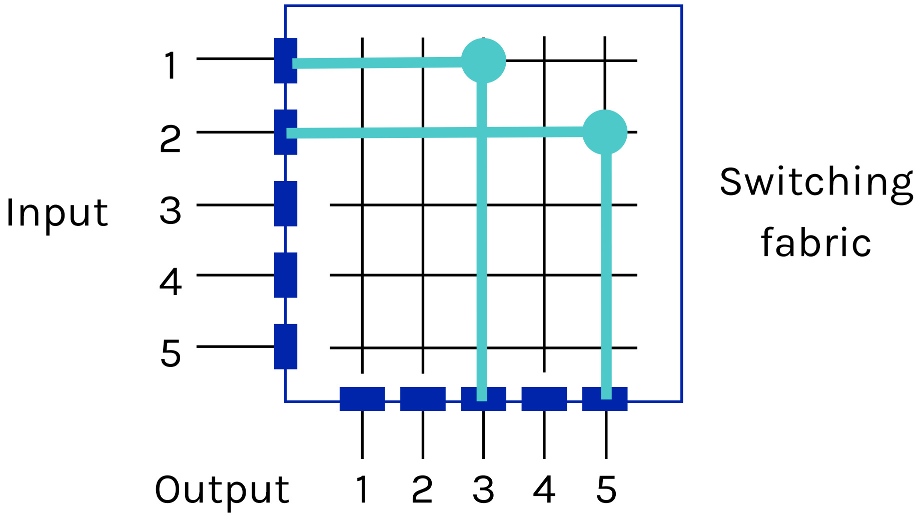
Inside a repeater



All inputs are connected; then amplified before going out

Inside a switch

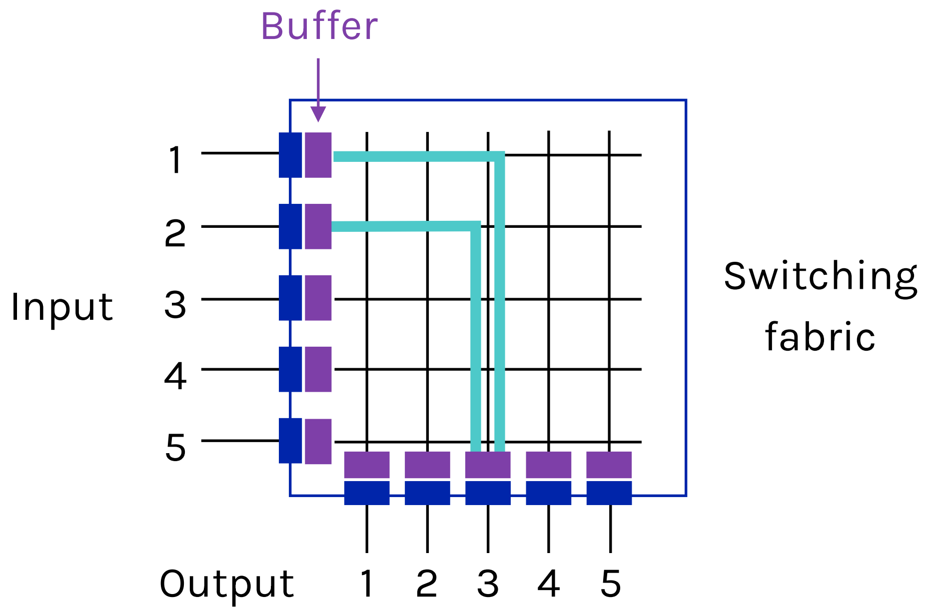
Uses frame addresses (MAC addresses in Ethernet) to connect input port to the right output port; multiple frames may be switched in parallel



Parallel switching: 1→3, 2→5

Frame buffering

Buffers needed when multiple inputs send to one output



Parallel switching to the same destination: 1→3, 2→3

Advantages of switches

Switches have replaced the shared cable of classic Ethernet

- Convenient to run wires to one location
- More reliable; wire cut is not a single point of failure that is hard to find

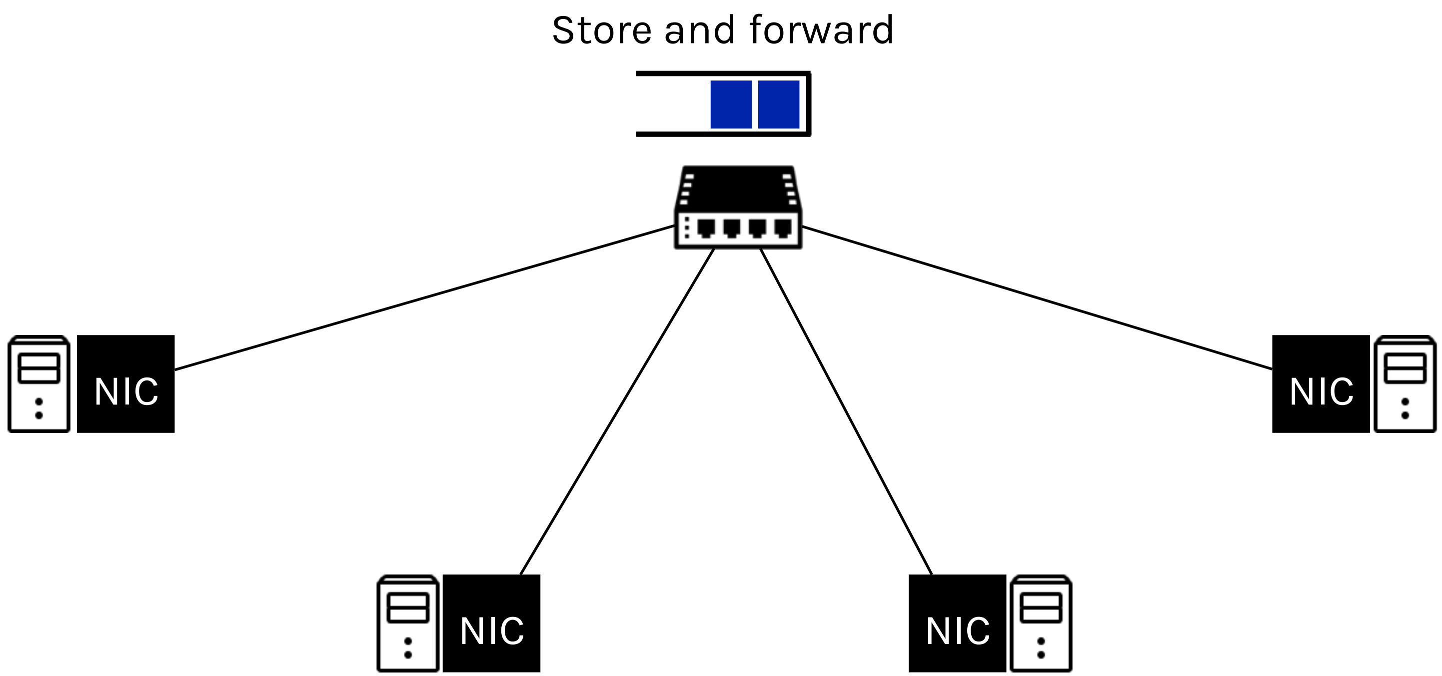
Switches offer scalable performance

- For example, 100 Mbps per port instead of 100 Mbps for all nodes of shared cable or hub

Modern switches support 400 Gbps per port and an aggregate throughput of 10 Tbps

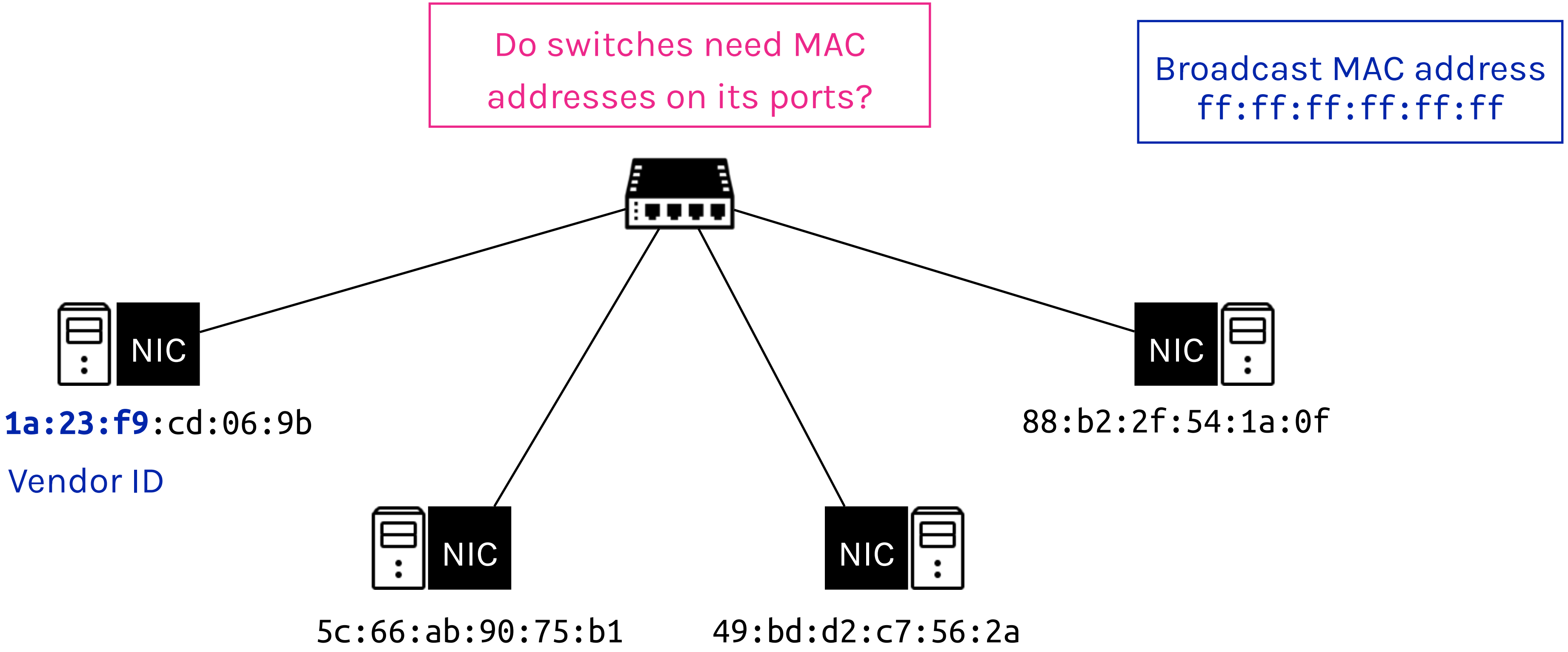
https://www.juniper.net/us/en/solutions/400g.html

Switched Ethernet

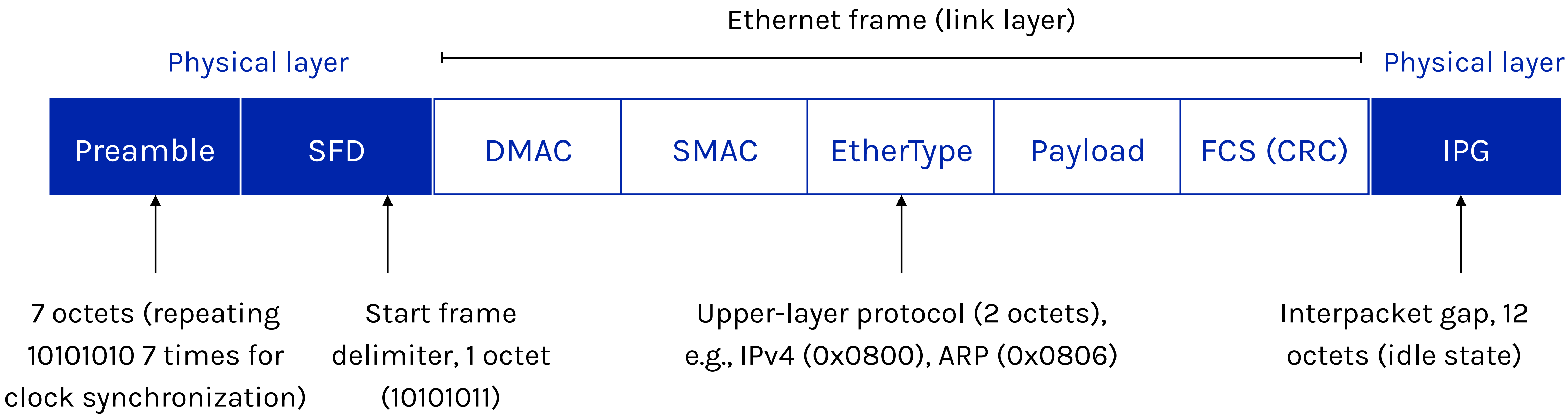


Ethernet MAC address

6-byte long, unique among all network adapters, managed by IEEE



Ethernet frame structure



Ethernet efficiency

$$\text{Protocol overhead} = \frac{\text{Total size - Payload size}}{\text{Total size}}$$

$$\text{Protocol efficiency} = \frac{\text{Payload size}}{\text{Total size}}$$

1460 if excluding TCP & IP headers

$$\text{Ethernet efficiency} = \frac{1500}{1500 + 7 + 1 + 6 + 6 + 2 + 4 + 12}$$

$$= 97.53\%$$

What about using Jumbo frames?

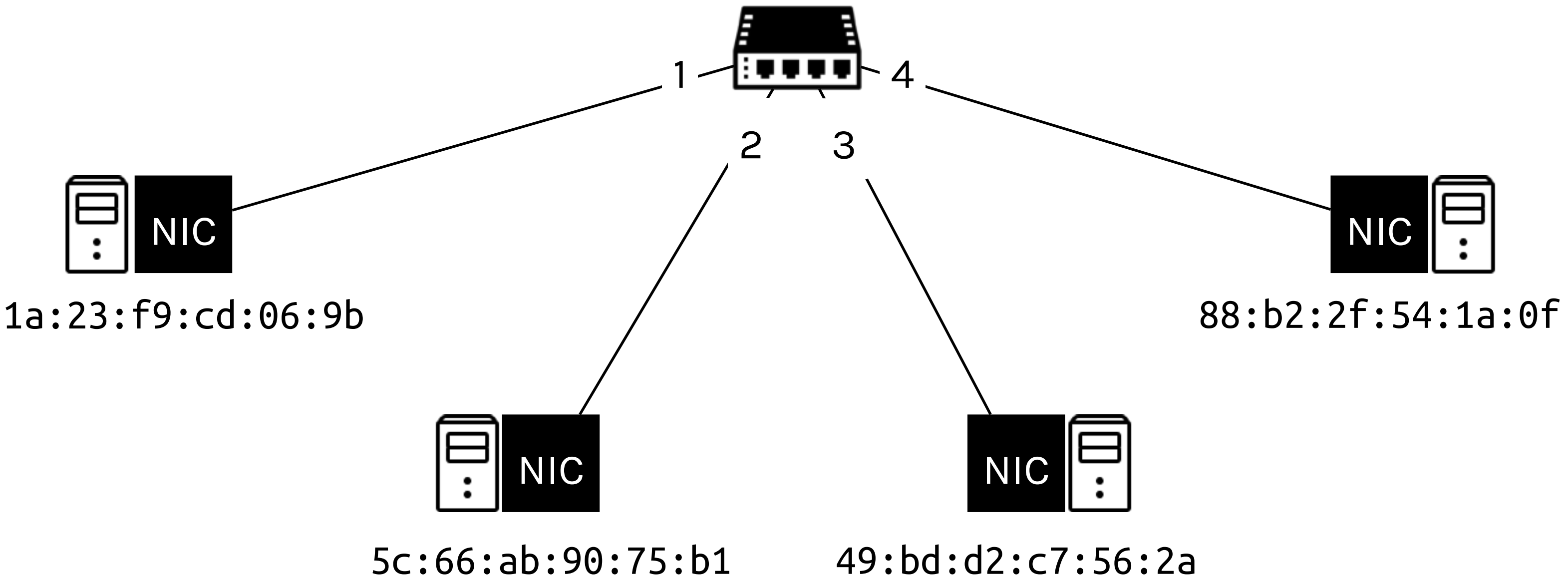
https://en.wikipedia.org/wiki/Jumbo_frame

Link layer forwarding

Switches forward/broadcast/drop frames based on a forwarding table

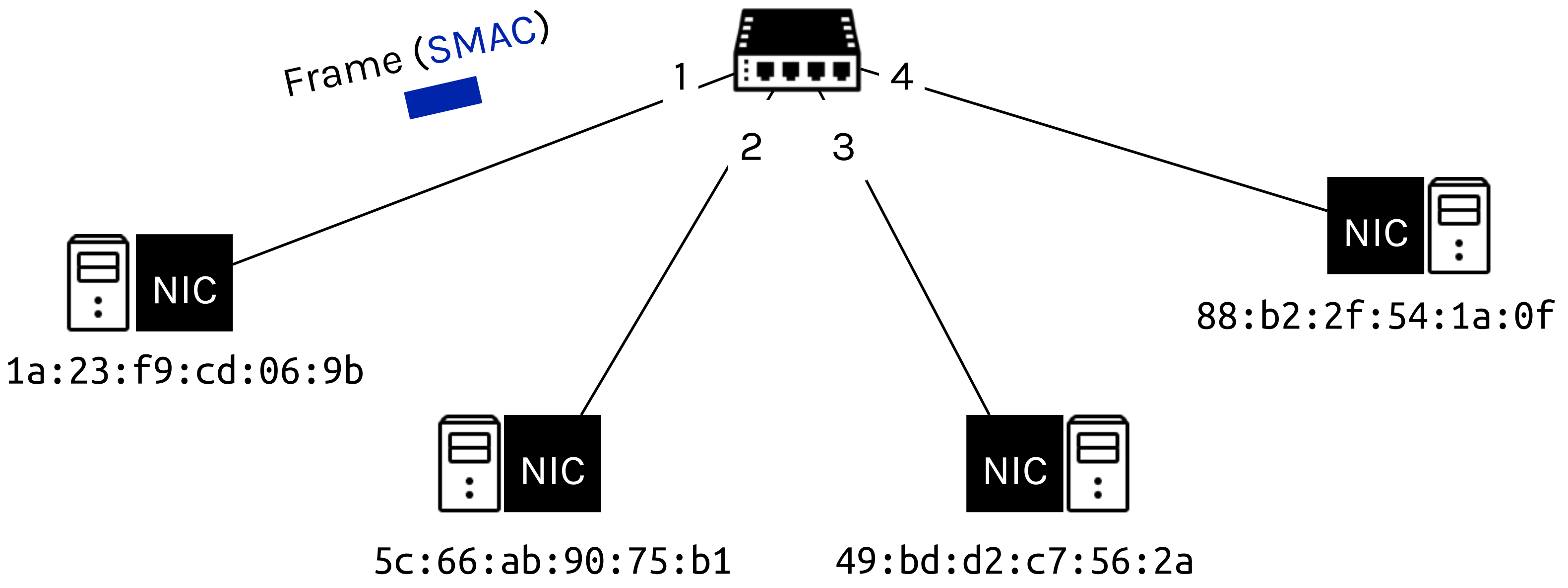
DMAC	Port	Time
88:b2:2f:54:1a:0f	4	8:34
5c:66:ab:90:75:b1	2	9:12

How to construct this table?



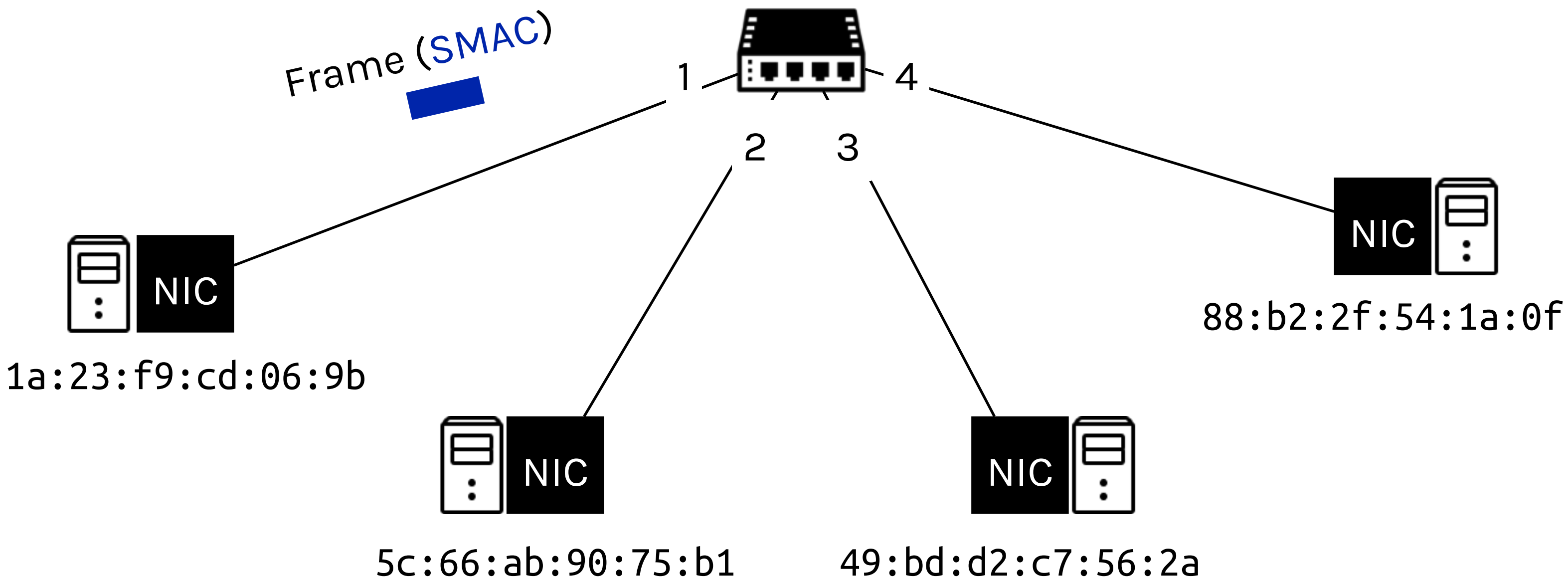
Learning

DMAC	Port	Time
88:b2:2f:54:1a:0f	4	8:34
5c:66:ab:90:75:b1	2	9:12



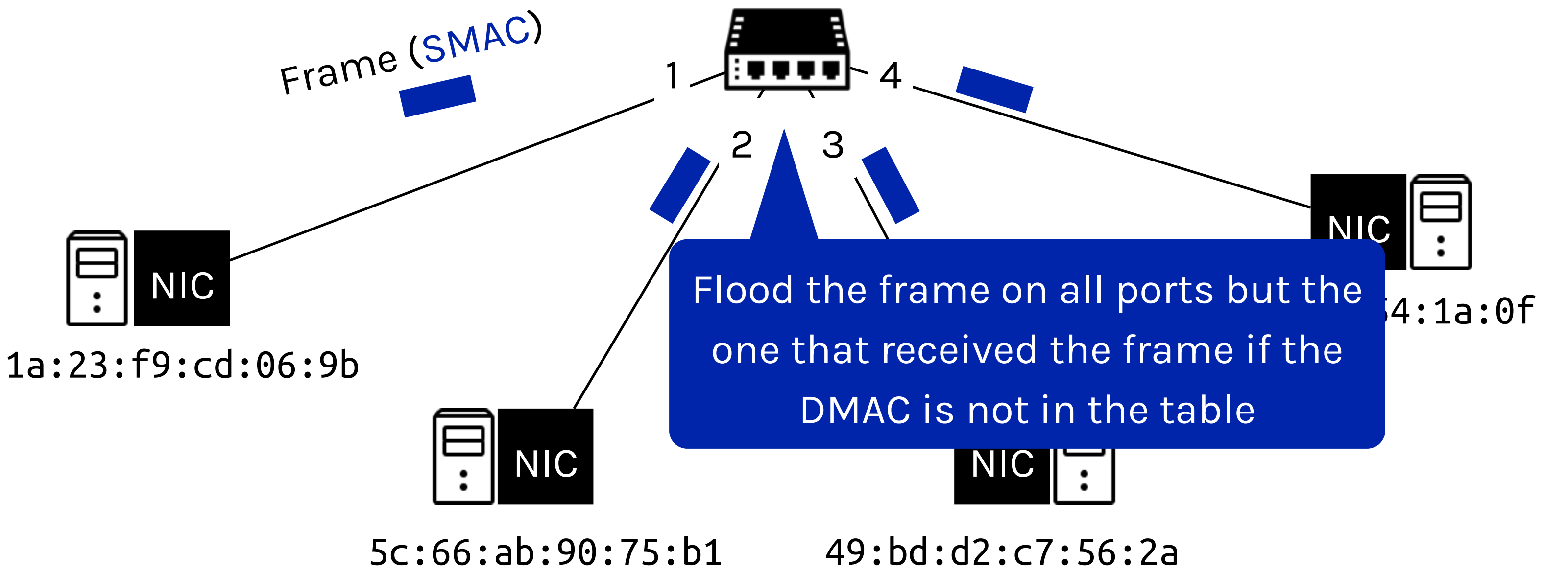
Learning

DMAC	Port	Time
88:b2:2f:54:1a:0f	4	8:34
5c:66:ab:90:75:b1	2	9:12
1a:23:f9:cd:06:9b	1	10:00



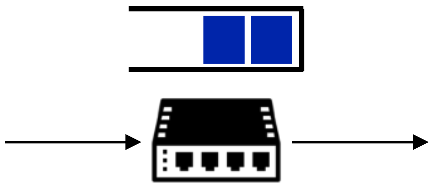
Backward learning

DMAC	Port	Time
88:b2:2f:54:1a:0f	4	8:34
5c:66:ab:90:75:b1	2	9:12
1a:23:f9:cd:06:9b	1	10:00



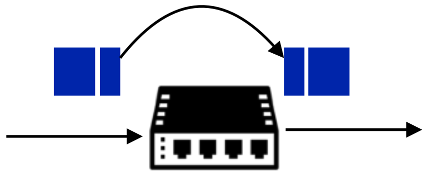
Store-and-forward vs. cut-through

Store-and-forward



Packets are received in full, buffered, and forwarded onto the output link

Cut-through

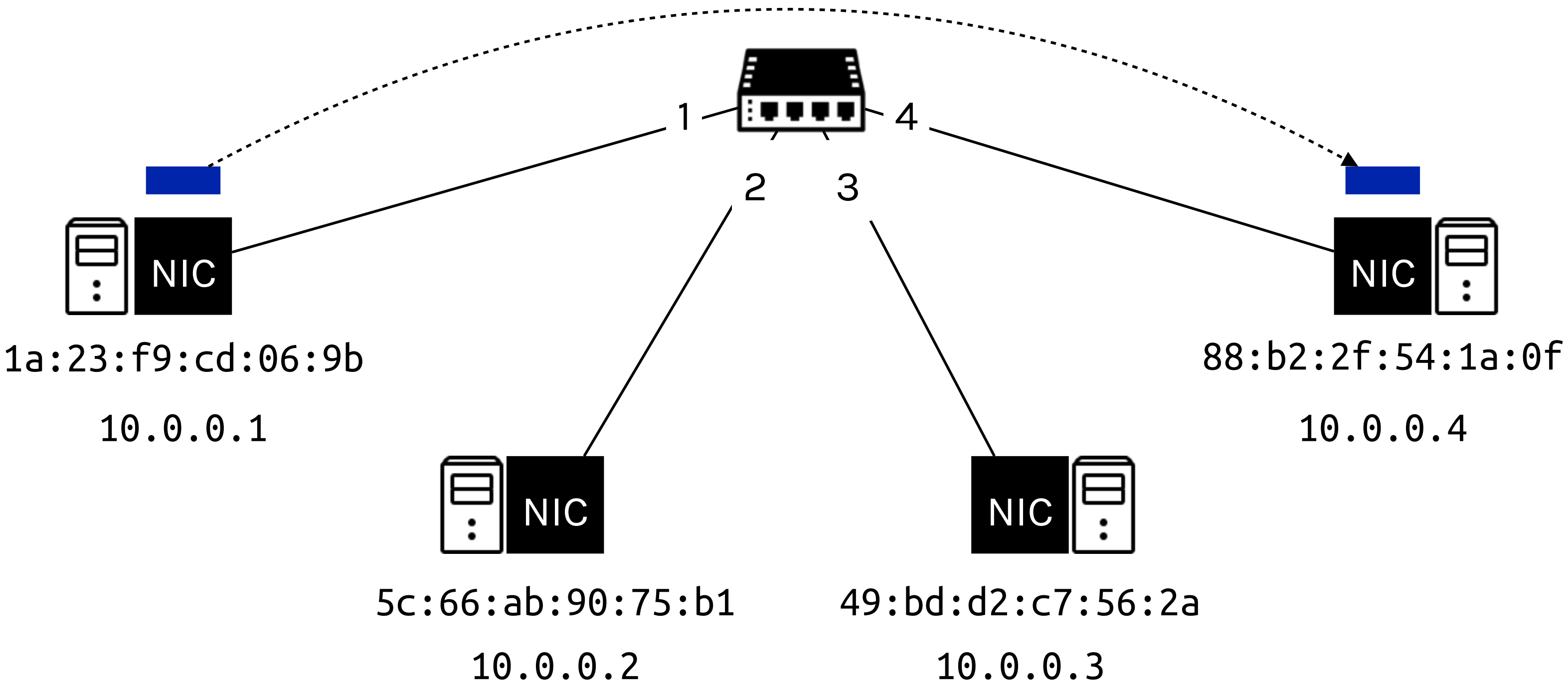


Once lookup is done, packet receiving and sending happen at the same time

What are the pros and cons of each approach?

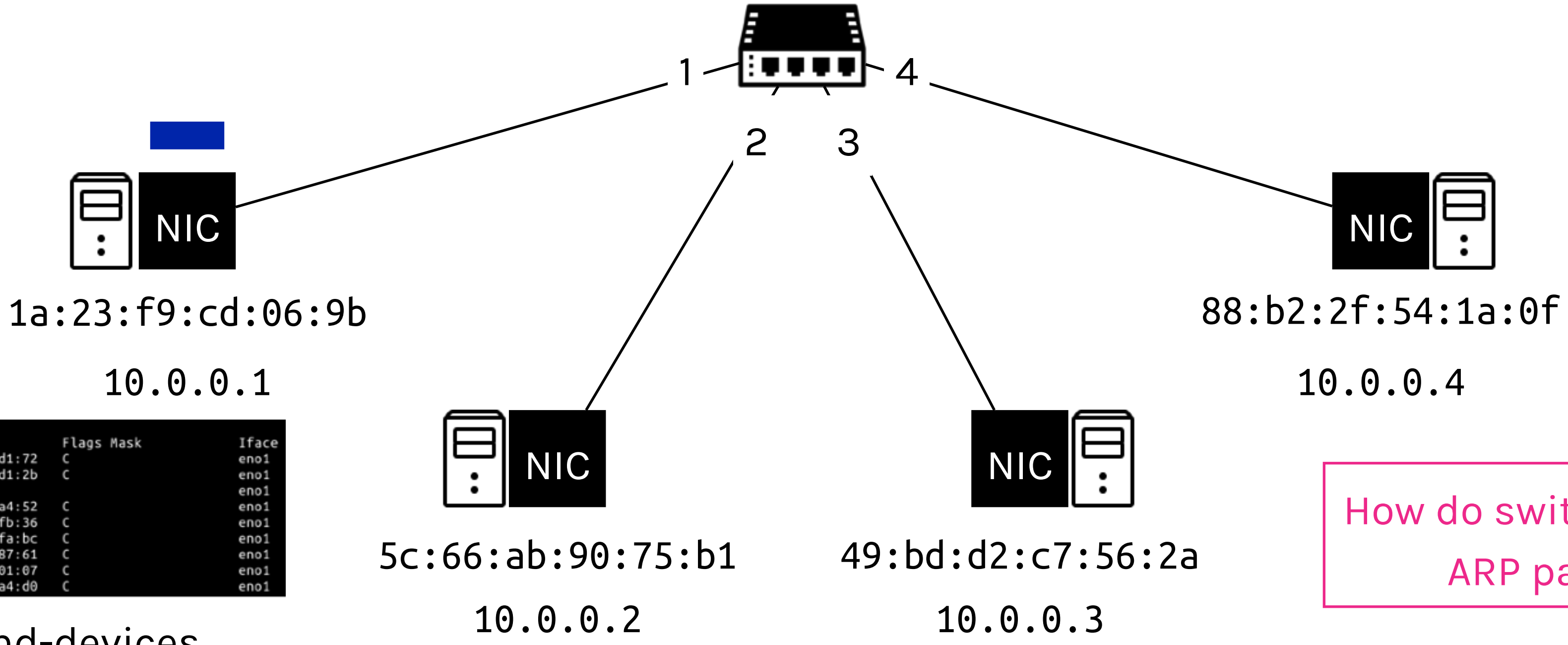
How to know the destination MAC address?

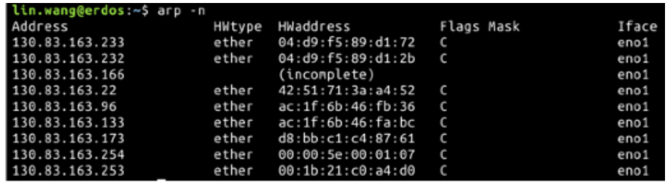
Assume we want to send some data from `10.0.0.1` to `10.0.0.4`. We know switches work with MAC addresses, but how does `10.0.0.1` get to know the MAC address of `10.0.0.4`?



Address Resolution Protocol (ARP)

ARP query	Whoever has the IP of `10.0.0.4`, please respond to me
ARP response	That is me, and here is my MAC address `88:b2:2f:54:1a:0f`





```
lin.wang@erdos:~$ arp -n
Address                  HWtype  HWaddress           Flags Mask            Iface
130.83.163.233           ether   04:d9:f5:89:d1:72   C                     eno1
130.83.163.232           ether   04:d9:f5:89:d1:2b   C                     eno1
130.83.163.166                   (incomplete)                              eno1
130.83.163.22            ether   42:51:71:3a:a4:52   C                     eno1
130.83.163.96            ether   ac:1f:6b:46:fb:36   C                     eno1
130.83.163.133           ether   ac:1f:6b:46:fa:bc   C                     eno1
130.83.163.173           ether   d8:bb:c1:c4:87:61   C                     eno1
130.83.163.254           ether   00:00:5e:00:01:07   C                     eno1
130.83.163.253           ether   00:1b:21:c0:a4:d0   C                     eno1
```

ARP table on end-devices

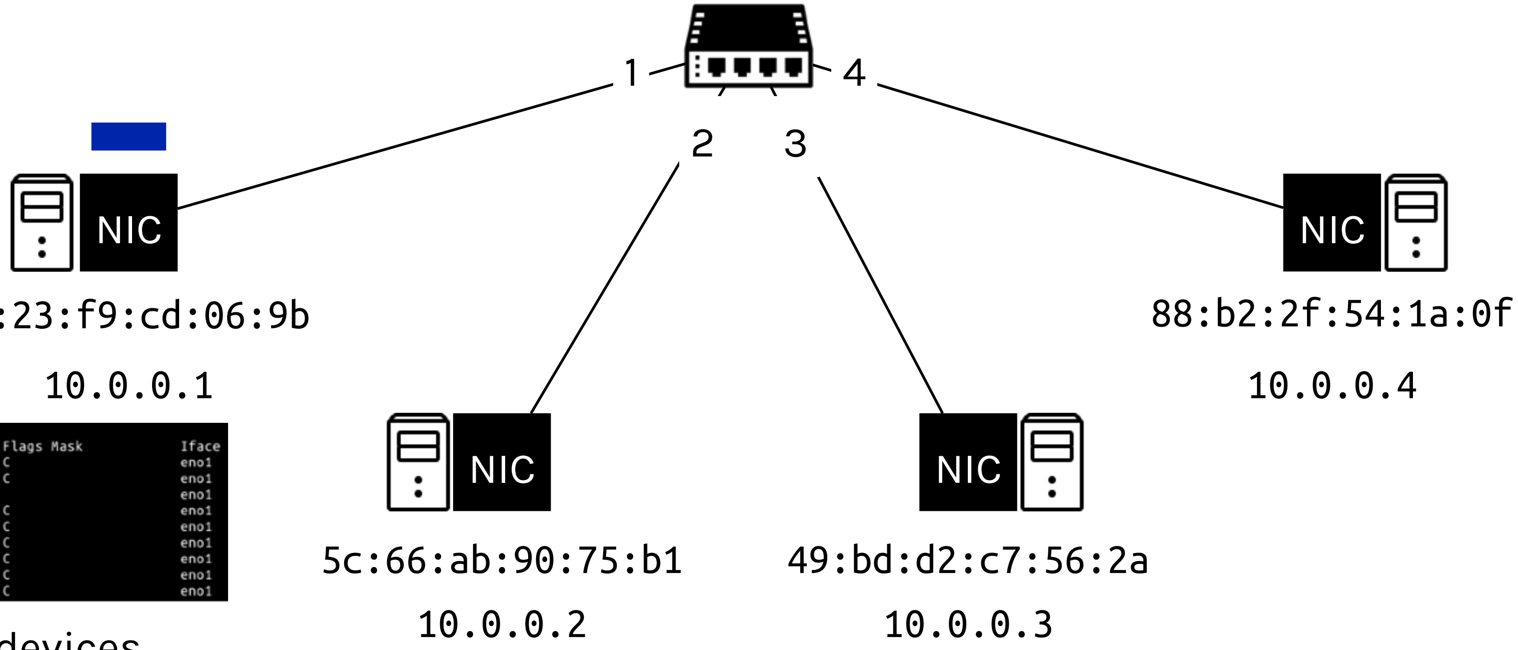
How do switches handle ARP packets?

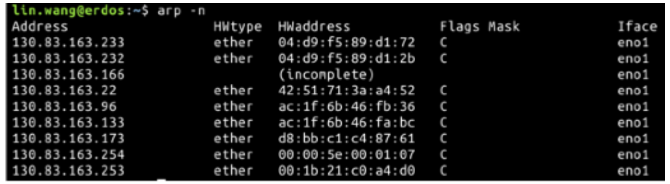
https://datatracker.ietf.org/doc/html/rfc826

How do switches handle ARP?

ARP query Switches broadcast the frame to all ports except the one received the frame

ARP response Switches forward the frame according to the DMAC





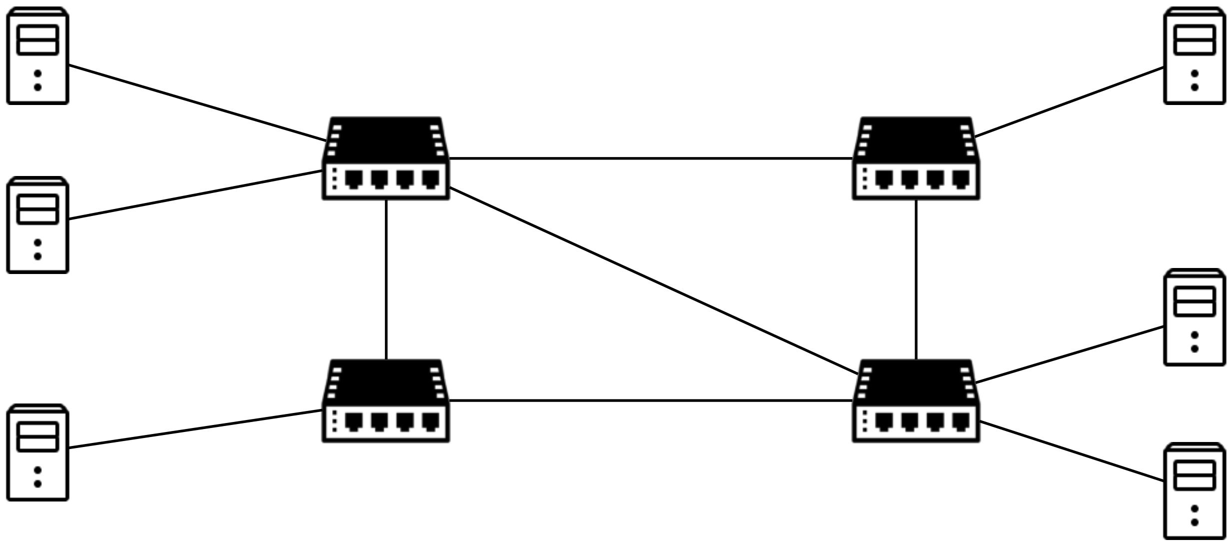
```
lin.wang@erdos:~$ arp -n
Address                  HWtype  HWaddress           Flags Mask            Iface
130.83.163.233           ether   04:d9:f5:89:d1:72   C                     eno1
130.83.163.232           ether   04:d9:f5:89:d1:2b   C                     eno1
130.83.163.166                   (incomplete)                              eno1
130.83.163.22            ether   42:51:71:3a:a4:52   C                     eno1
130.83.163.96            ether   ac:1f:6b:46:fb:36   C                     eno1
130.83.163.133           ether   ac:1f:6b:46:fa:bc   C                     eno1
130.83.163.173           ether   d8:bb:c1:c4:87:61   C                     eno1
130.83.163.254           ether   00:00:5e:00:01:07   C                     eno1
130.83.163.253           ether   00:1b:21:c0:a4:d0   C                     eno1
```

ARP table on end-devices

https://datatracker.ietf.org/doc/html/rfc826

A network of switches

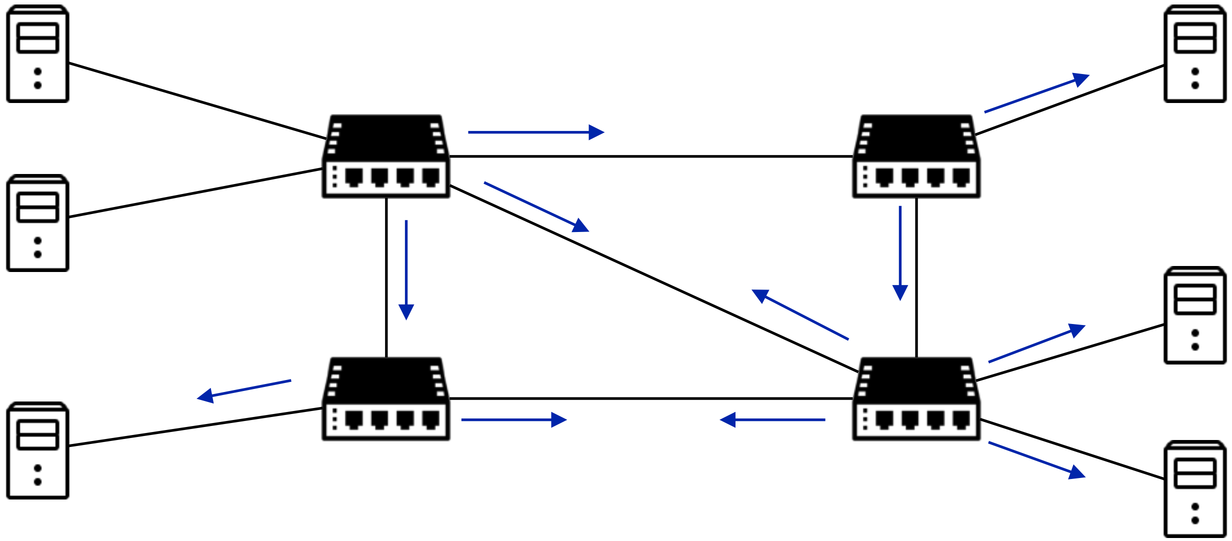
Multiple switches are used to interconnect a large number of end-devices



What are the problems?

When flooding meets loops

Loops are there for redundancy or simply a mistake

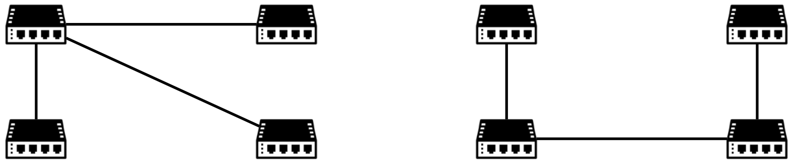


Each received frame, when being broadcast, will create at least two copies of the frame on the network, which can loop forever in the network

Spanning Tree Protocol (STP)

Key idea

- Reduce the network to one logical spanning tree (no loops, but reachable to all nodes) and disable links not on the spanning tree
- Upon failure, automatically rebuild a spanning tree



Radia Perlman

Distributed algorithms for STP

Rules of the distributed game

- All switches run the same algorithm
- They start with no information of others
- Operate in parallel and send messages
- Always search for the best solution

Ensures a highly robust solution

- Any topology, with no configuration
- Adaptes to link/switch failures

Algorhyme

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

Spanning tree algorithm

1. Elect a root node of the tree (switch with the lowest address)
2. Grow a tree as shortest distances from the root (using lowest address to break distance ties)
3. Turn off ports for forwarding if they are not on the spanning tree

Spanning tree algorithm details

Considered root	Hop count to root	Switch ID

Bridget protocol data unit (BPDU)

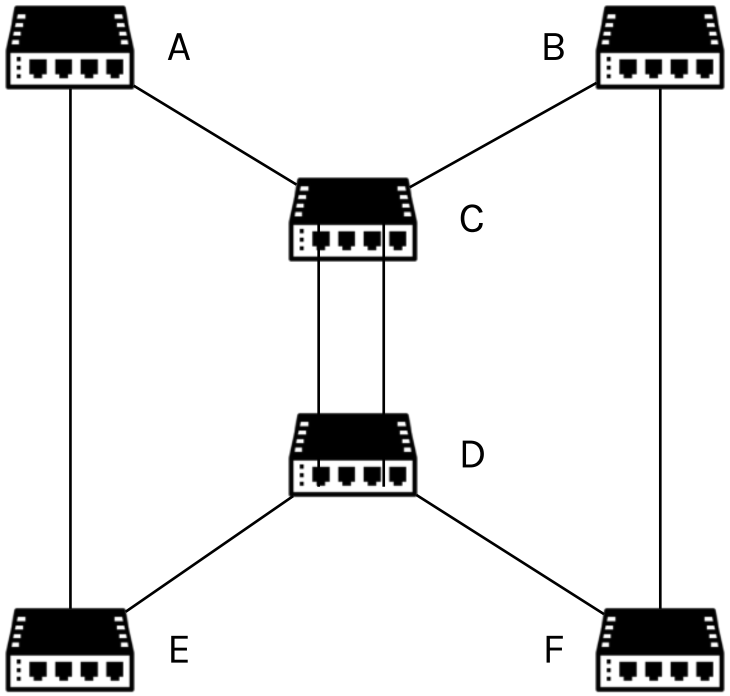
Initially

- Each switch proposes itself as root, i.e., send (X,0,X) on all its interfaces
- Upon receiving (Y,d,X), each switch checks if Y is a better root. If so, it considers Y as the new root, and floods updated message
- Switches compute their distance to the root, for each port: simply add one to the distance received; if shorter, flood
- Switches disable interfaces not on shortest path

Tie-breaking

- For BPDUs with equal distance, pick the one with the lower switch ID
- When receiving different BPDUs from a neighbor switch, keep the one with lowest port ID

Spanning tree example



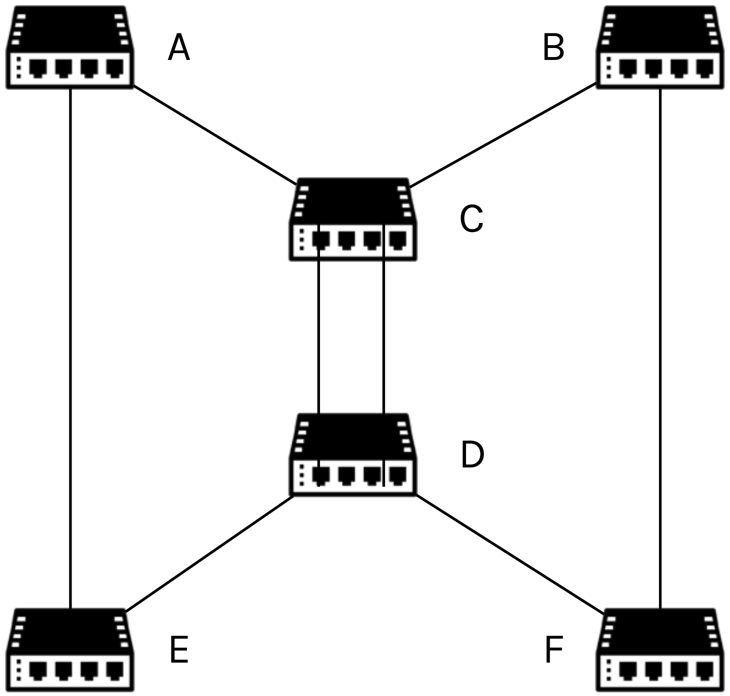
First round, sending

- A sends (A,0,A) to declare it is root
- B,C,D,E and F do likewise

First round, receiving

- A still thinks it is (A,0,A)
- B still thinks it is (B,0,B)
- C updates to (A,1,C)
- D updates to (C,1,D)
- E updates to (A,1,E)
- F updates to (B,1,F)

Spanning tree example



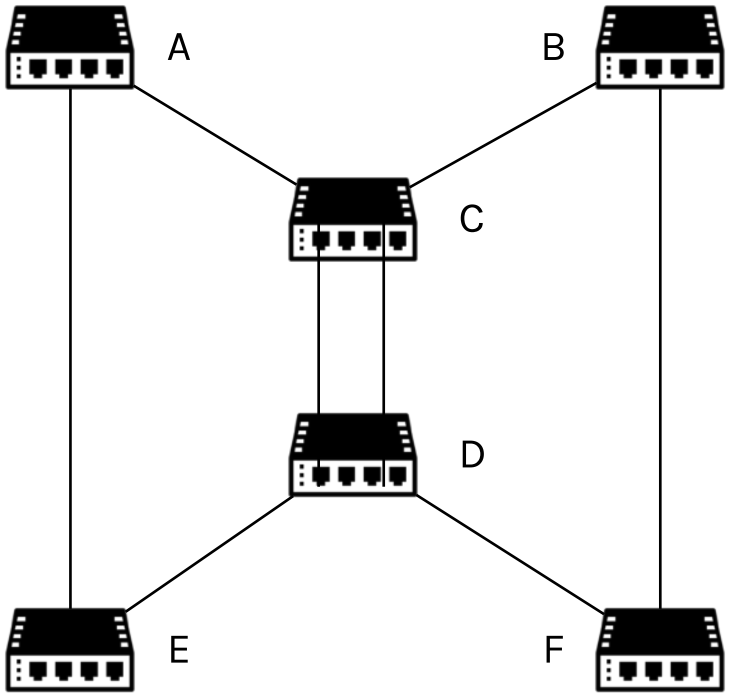
Second round, sending

- All nodes send their updated states

Second round, receiving

- A remains (A,0,A)
- B updates to (A,2,B) via C
- C remains (A,1,C)
- D updates to (A,2,D) via C
- E remains (A,1,E)
- F remains (B,1,F)

Spanning tree example



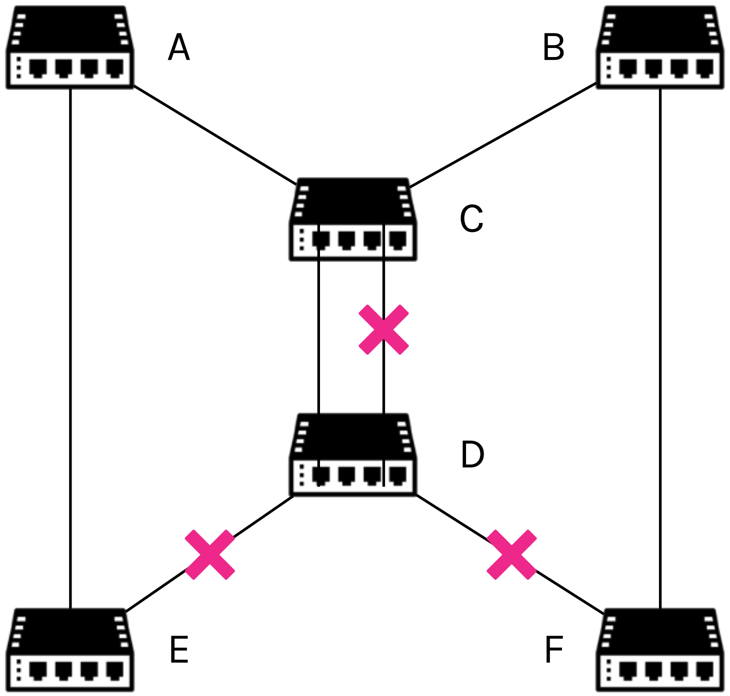
Third round, sending

- All nodes send their updated states

Third round, receiving

- A remains (A,0,A)
- B remains (A,2,B) via C
- C remains (A,1,C)
- D remains (A,2,D) via C
- E remains (A,1,E)
- F updates to (A,1,F) via B

Spanning tree example



Steady-state reached

- Notes turn off links that are not on the spanning tree

Algorithm continues to run

- Adaptes by timing out information: root continuously announcing itself; upon timeout, claim to be the root
- For example: If A fails, other nodes forget it, and B will become the new root

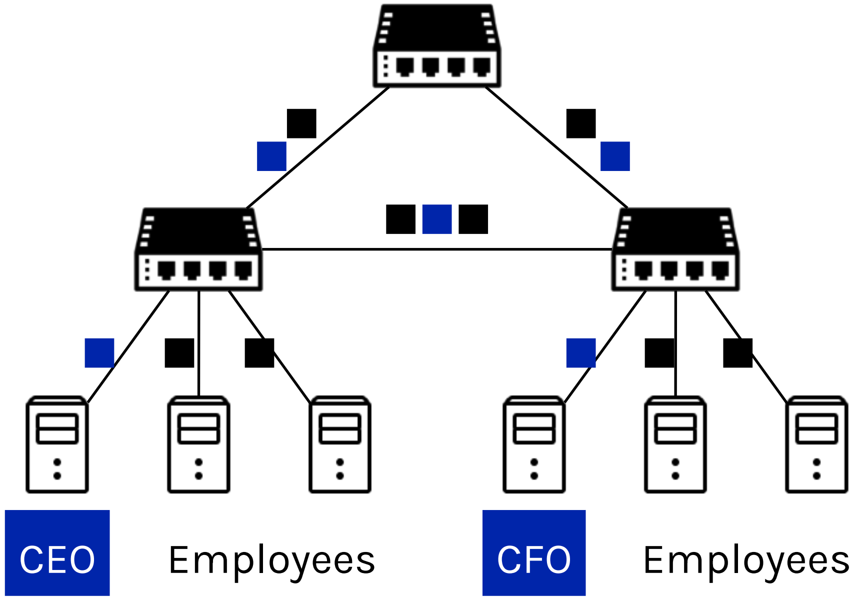
STP is susceptible to attacks.

Think about why!

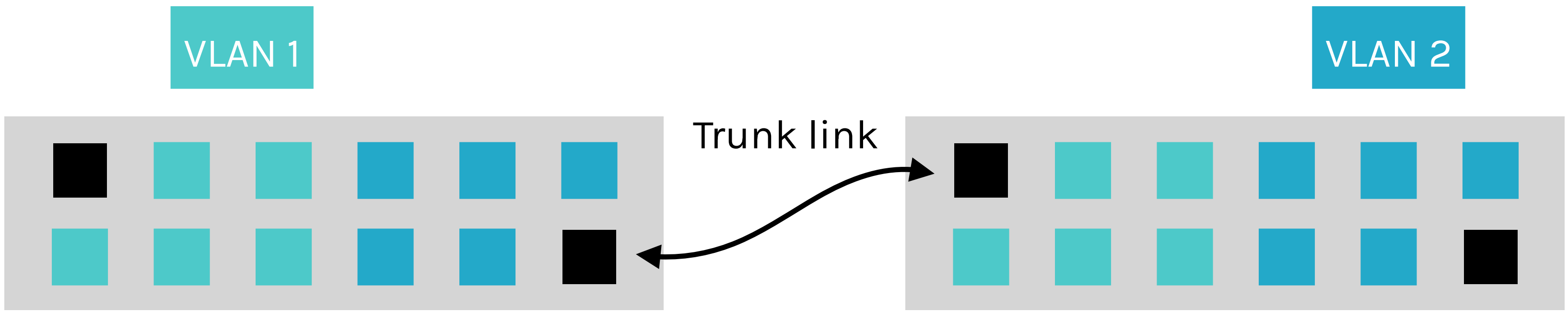
Traffic isolation on Ethernet

Broadcast packets cannot be localized and can cause broadcast storm in the network

Hard user management: A user has to be connected to the particular switch in order to isolate its traffic



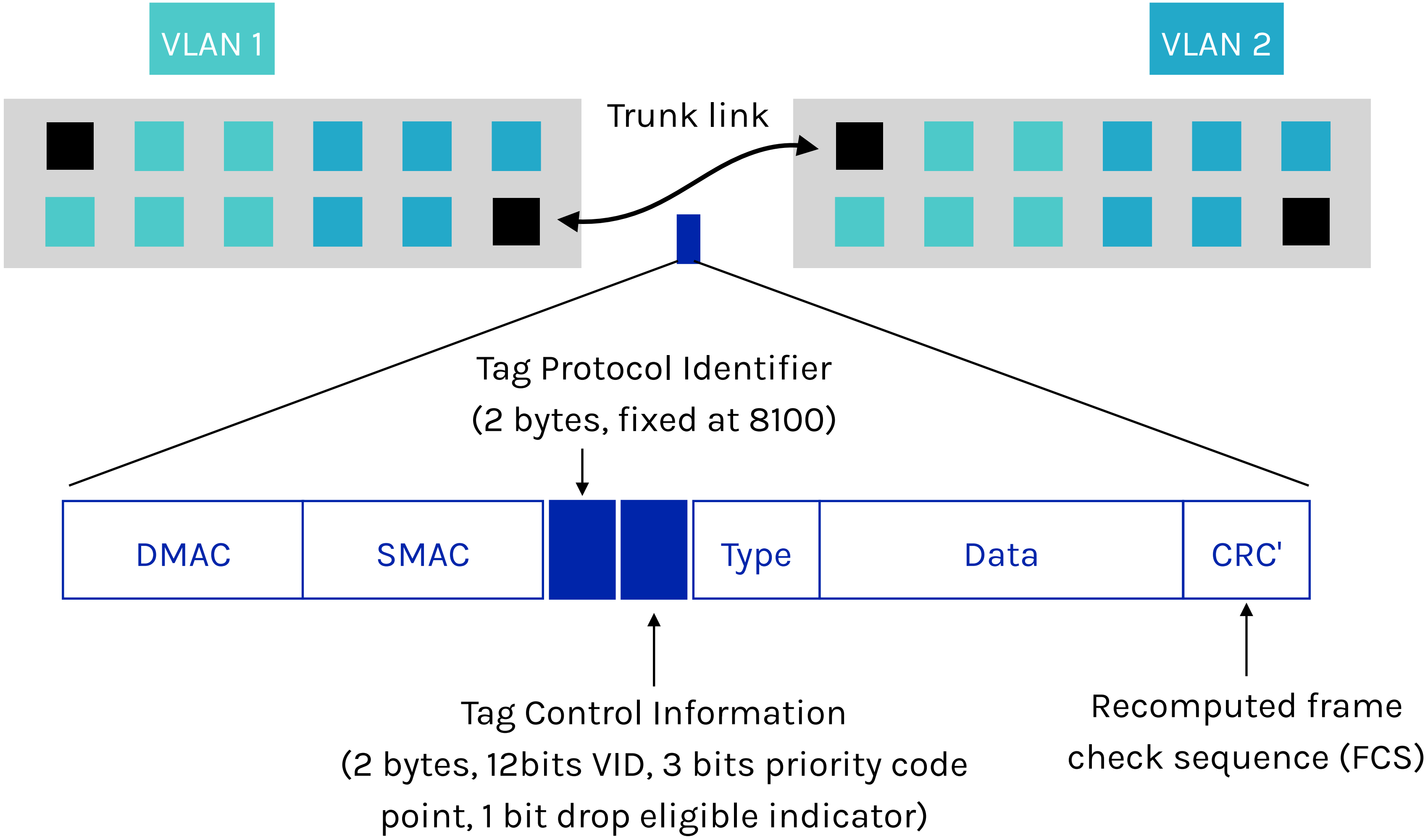
VLAN



- Partition the ports on switches into subsets and assign them to VLANs
- Ports in the same VLAN form a broadcast domain; ports on different VLANs are routed via an internal router within the switch
- Switches are connected on trunk links that belong to all VLANs
- Per-VLAN learning of the forwarding table
- Per-VLAN spanning tree construction

How does a switch know which VLAN a frame belongs to?

VLAN tag


VLAN 1
VLAN 2
Trunk link
Tag Protocol Identifier
(2 bytes, fixed at 8100)
DMAC
SMAC
Type
Data
CRC'
Tag Control Information
(2 bytes, 12bits VID, 3 bits priority code
point, 1 bit drop eligible indicator)
Recomputed frame
check sequence (FCS)

Other issues of Ethernet

Broadcast storm

- ARP requests, MAC addresses that have not been learned

Limited switch forwarding table size

- A limited number of entries can be cached

Limited isolation with VLAN

- Max. 4096 VLANs are possible (only 12 bits for the VID)

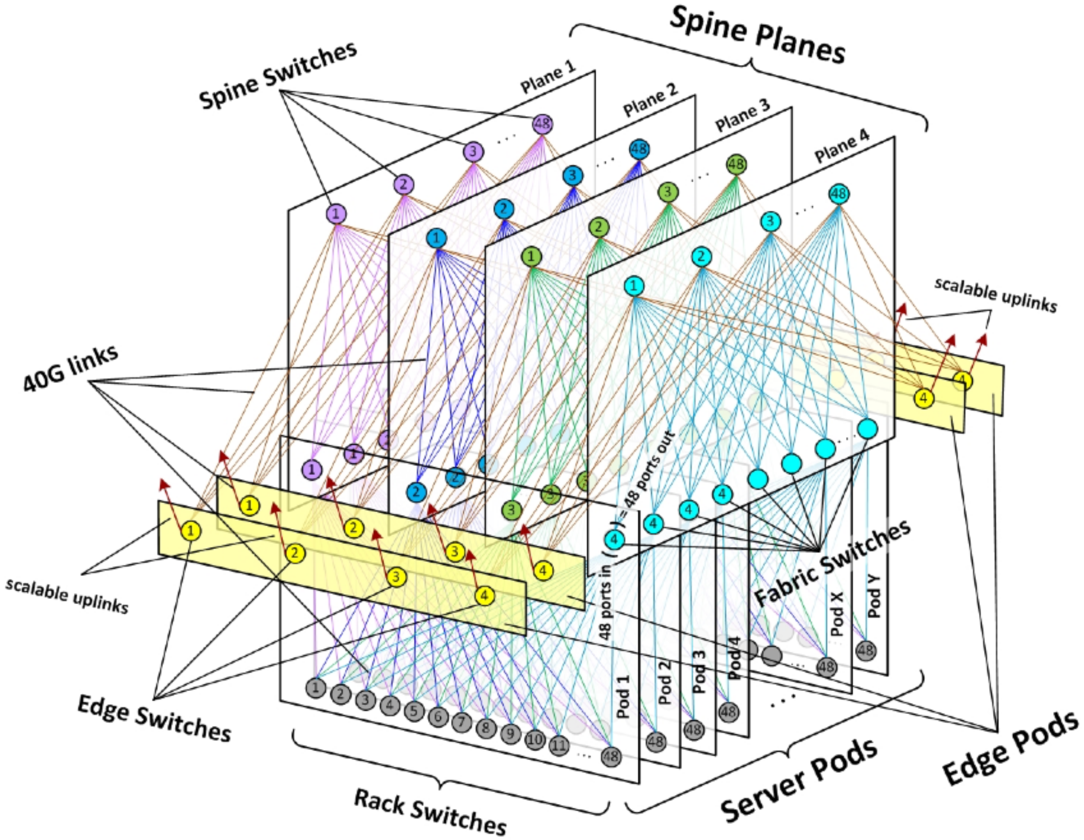
Security issues: packet sniffing, ARP spoofing attack, MAC flooding attack

How to scale the network up?

A couple of servers

Up to millions of servers in a typical data center

Data center networking



Scaling the link layer

Fundamentally, it is hard to scale distributed algorithms

- Exacerbated when failures become common
- Nodes go down, have to run spanning tree again…
- If nodes go down faster than spanning tree evolves, we get race conditions
- If they do not, we may still be losing paths and wasting resources

Can we replace the distributed algorithm with a centralized control?

Software defined networking (SDN)

Core idea: stop being a distributed system

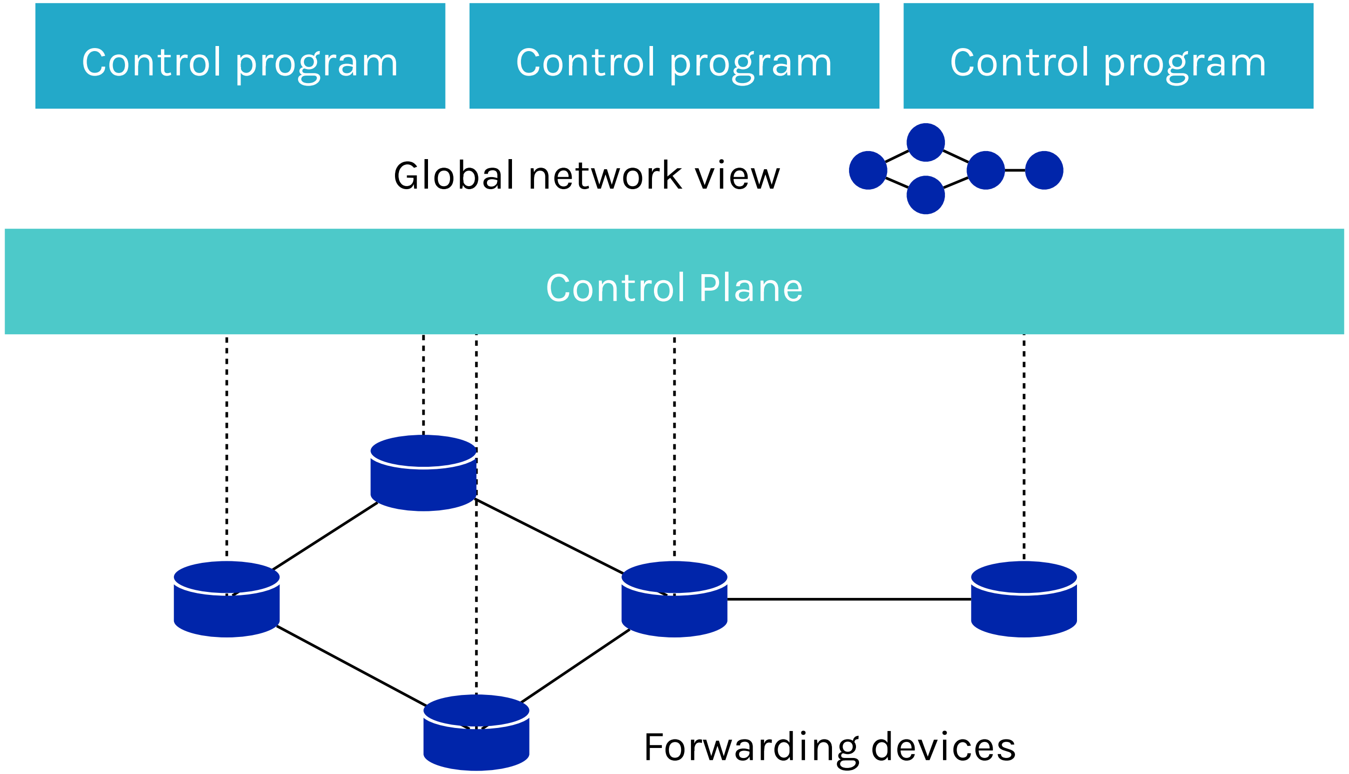
- Centralize the operation of the network
- Create a "controller" that manages the network

Push new code, state, and configuration from "controller" to switches

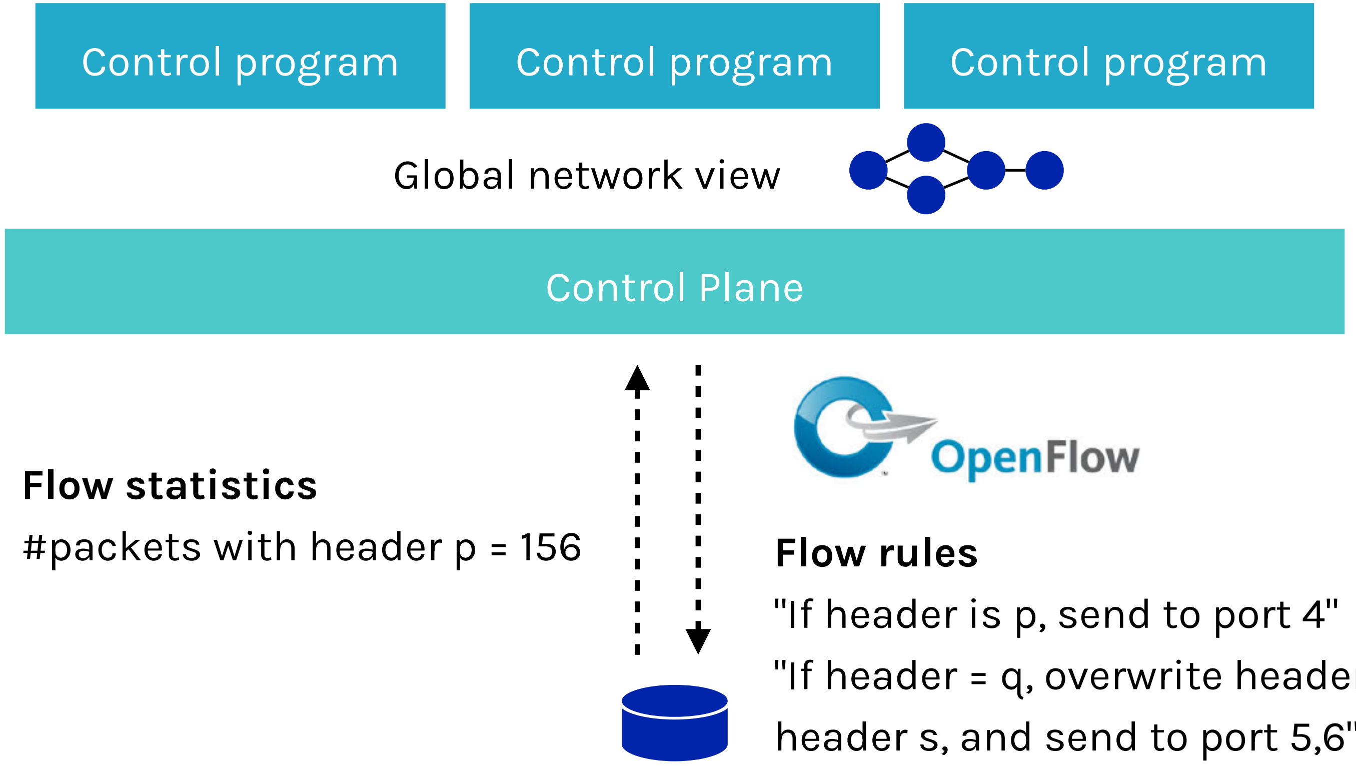
- Run link state with a global view of the network rather than in a distributed fashion
- Allows for global policies to be enforced
- Can resolve failures in more robust, faster manners

But there is no free lunch to that…

SDN architecture



OpenFlow



Flow statistics

#packets with header p = 156

Flow rules

"If header is p, send to port 4"

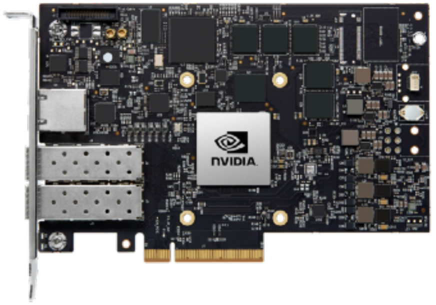
"If header = q, overwrite header with r, add header s, and send to port 5,6"

"If header = ?, send to the controller"

SDN benefits

Simplified network management

- From distributed protocols to centralized programs
- Encode code reuse and libraries
- Standardize and simplify deployment of rules to switches
- Better scalability due to simplified management



Accelerated innovations

- New ideas for networking
- Combined with programmable data plane

Summary

Multiple access

- TDM/FDM
- Random access: CSMA/CD
- Wireless networks and MACA

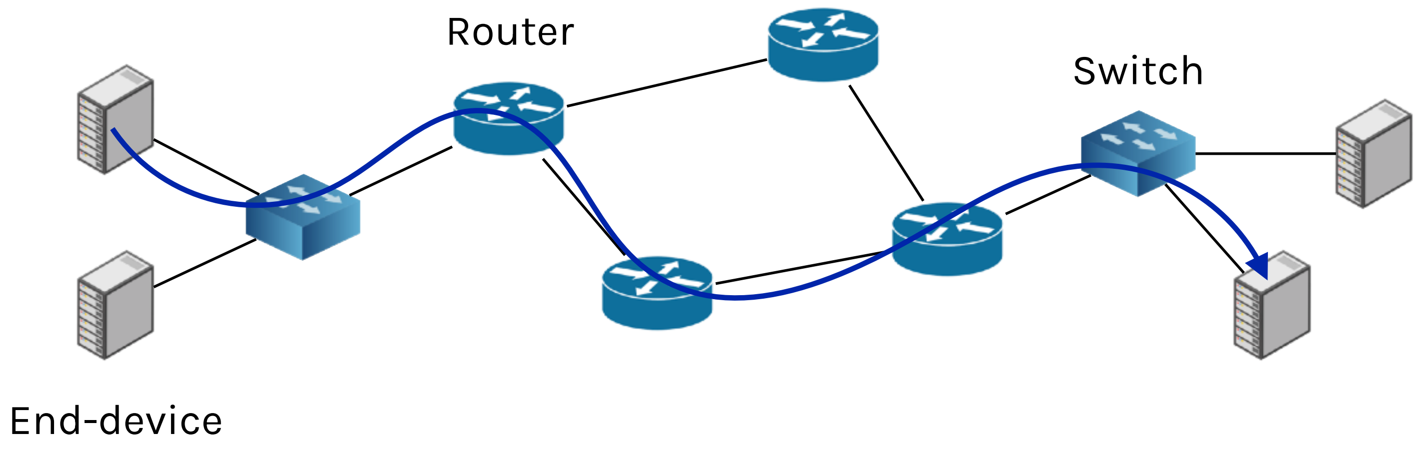
Ethernet and switching

- Switch architecture
- Ethernet switching, ARP
- STP, VLAN

Software-defined networking

- Scaling the link layer
- Centralized network control
- OpenFlow

Next time: network layer



How to route a packet from one end-device to another across networks?

Further reading material

Andrew S. Tanenbaum, David J. Wetherall. Computer Networks (5th edition).

- Chapter 4: The Medium Access Control Sublayer

Larry Peterson, Bruce Davie. Computer Networks: A Systems Approach.

- Section 2.6: Multi-Access Networks
- Section 2.7: Wireless Networks
- Section 3.1: Switching Basics
- Section 3.2: Switched Ethernet

Nick Feamster, Jennifer Rexford, Ellen Zegura. The Road to SDN: An Intellectual History of Programmable Networks. ACM SIGCOMM CCR, vol. 44(2), pp. 87-98.