



南京大學

NANJING UNIVERSITY

链路层和局域网



Outline

- Introduction
- Error detection, correction
- Multiple access protocols
- LANs
- Link virtualization: MPLS
- Data center networking
- A day in the life of a web request



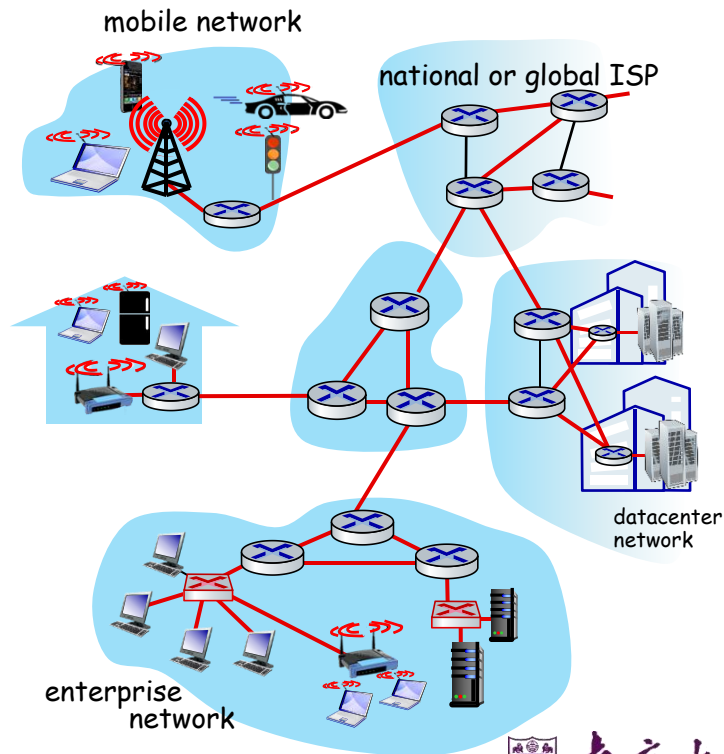


Link layer: introduction

terminology:

- hosts, routers: **nodes**
- communication channels that connect **adjacent** nodes along communication path: **links**
 - wired , wireless
 - LANs
- layer-2 packet: **frame**, encapsulates datagram

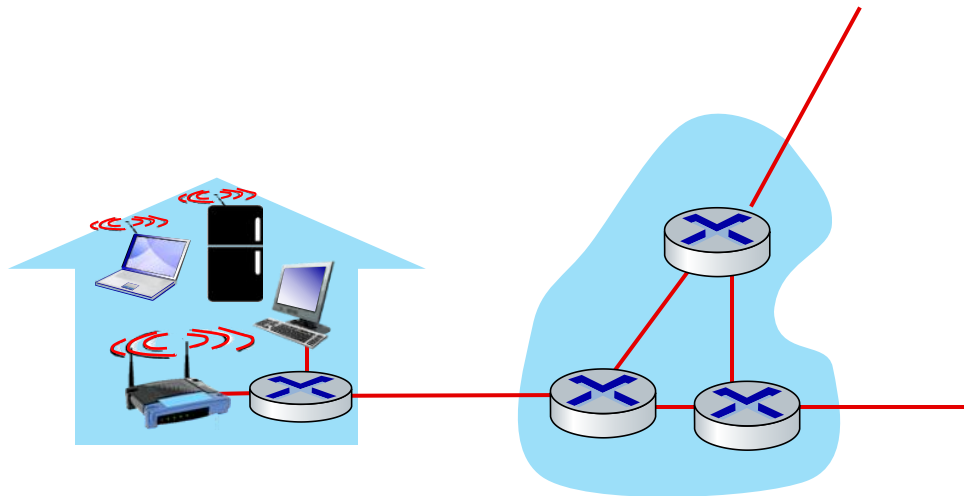
link layer has responsibility of transferring datagram from one node to **physically adjacent** node over a link





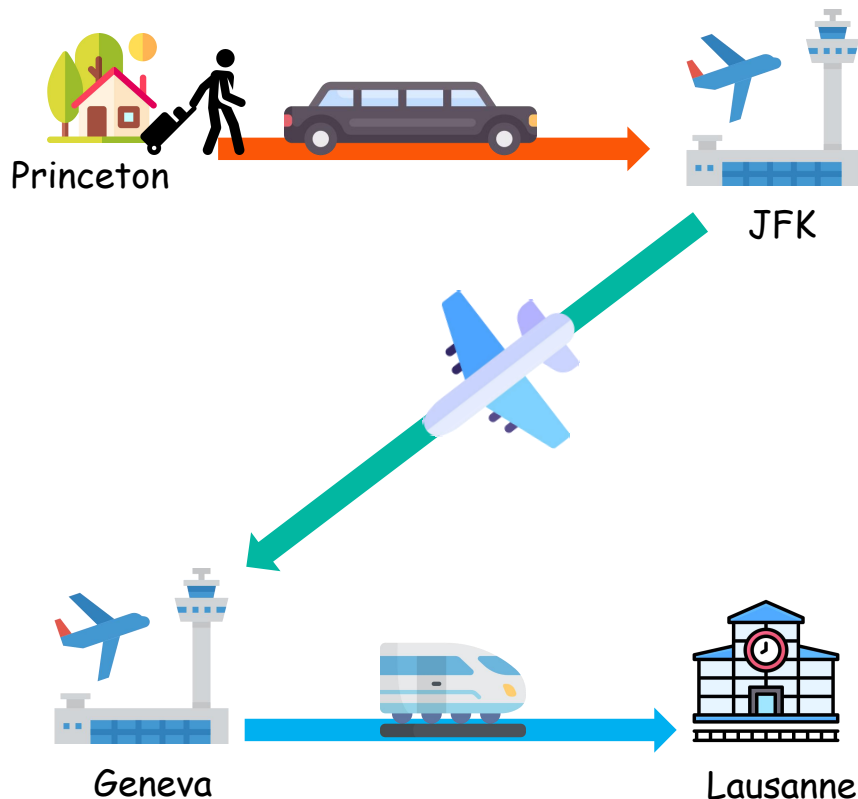
Link layer: context

- datagram transferred by **different link protocols** over different links:
 - e.g., WiFi on first link, Ethernet on next link
- each link protocol provides different services
 - e.g., **may or may not** provide reliable data transfer over link





Transportation analogy



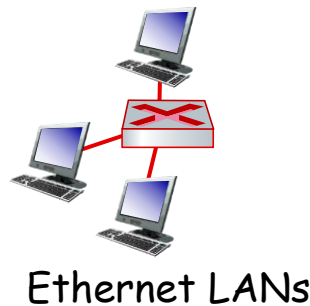
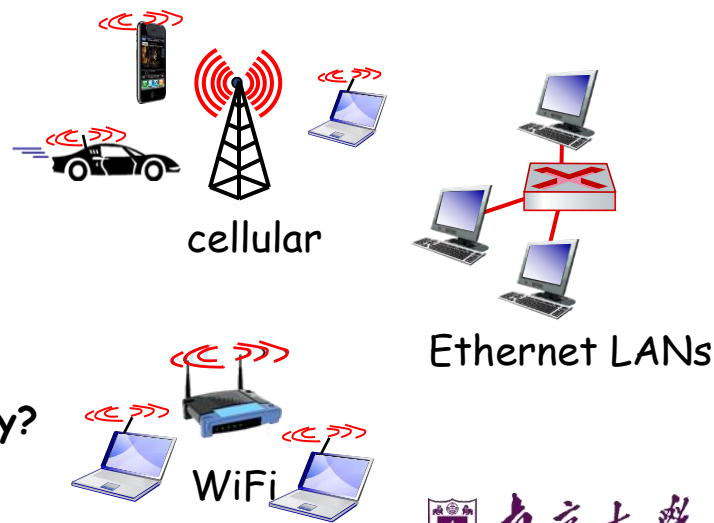
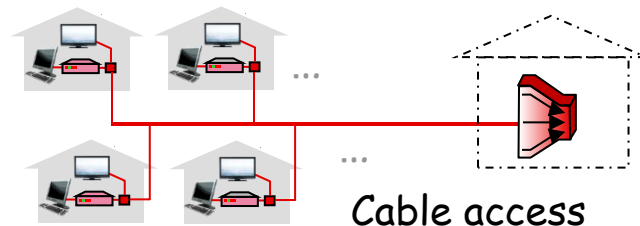
transportation analogy:

- trip from Princeton to Lausanne
 - limo: Princeton to JFK
 - plane: JFK to Geneva
 - train: Geneva to Lausanne
- tourist = **datagram**
- transport segment = **communication link**
- transportation mode = **link-layer protocol**
- travel agent = **routing algorithm**



Link layer: services

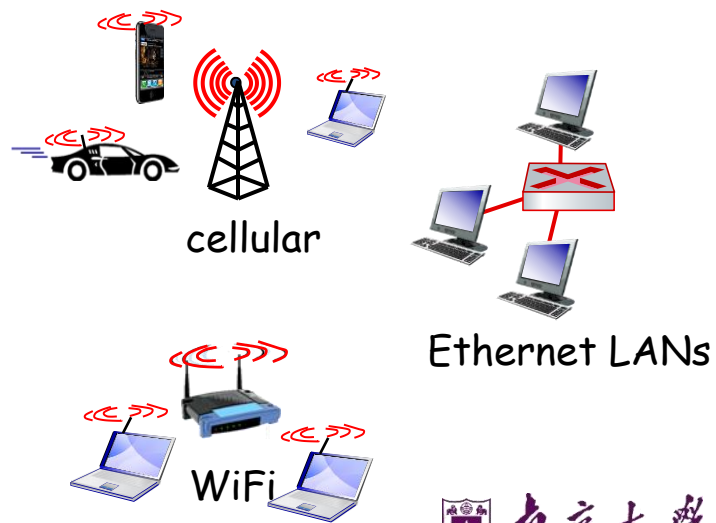
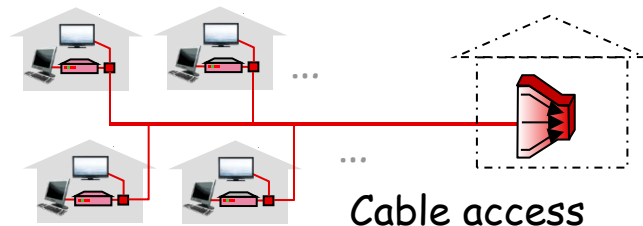
- **framing, link access:**
 - encapsulate datagram into frame, adding header, trailer
 - channel access if shared medium
 - "MAC" addresses in frame headers identify source, destination (different from IP address!)
- **reliable delivery between adjacent nodes**
 - we already know how to do this!
 - seldom used on low bit-error links
 - wireless links: high error rates
 - ✓ Q: why both link-level and end-end reliability?





Link layer: services (more)

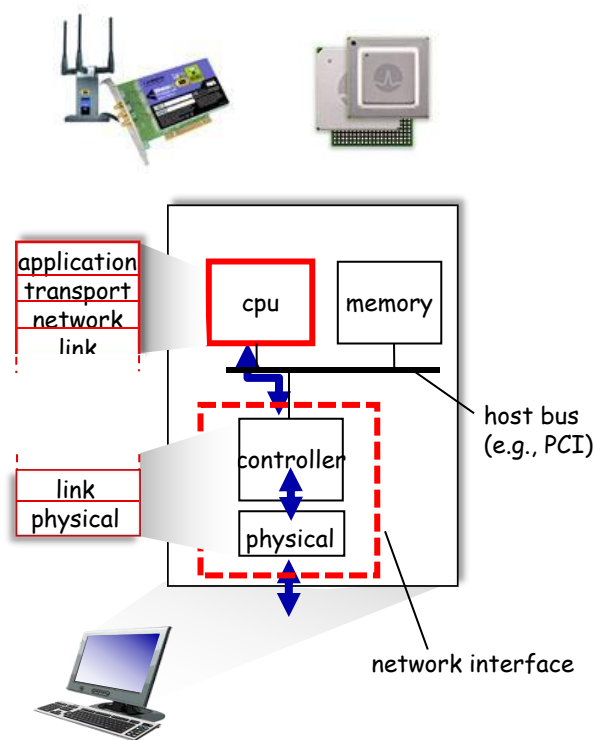
- **flow control:**
 - pacing between adjacent sending and receiving nodes
- **error detection:**
 - errors caused by signal attenuation, noise.
 - receiver detects errors, signals retransmission, or drops frame
- **error correction:**
 - receiver identifies and corrects bit error(s) without retransmission
- **half-duplex and full-duplex:**
 - with half duplex, nodes at both ends of link can transmit, but not at same time





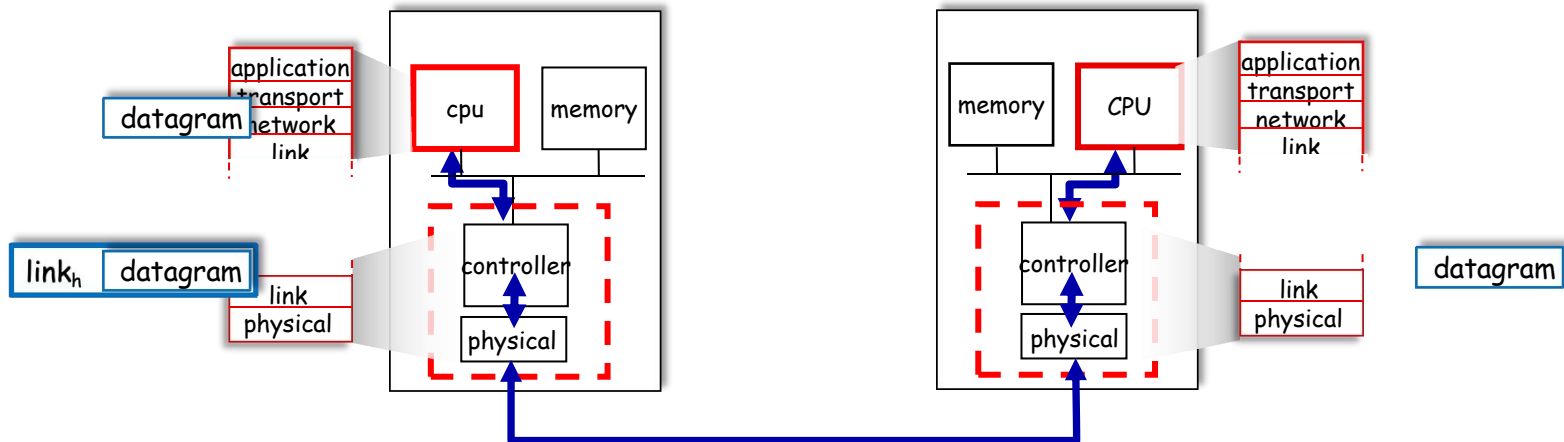
Host link-layer implementation

- in each-and-every host
- link layer implemented on-chip or in network interface card (NIC)
 - implements link, physical layer
- attaches into host's system buses
- combination of hardware, software, firmware





Interfaces communicating



sending side:

- encapsulates datagram in frame
- adds error checking bits, reliable data transfer, flow control, etc.

receiving side:

- looks for errors, reliable data transfer, flow control, etc.
- extracts datagram, passes to upper layer at receiving side



Outline

- Introduction
- Error detection, correction
- Multiple access protocols
- LANs
- Link virtualization: MPLS
- Data center networking
- A day in the life of a web request

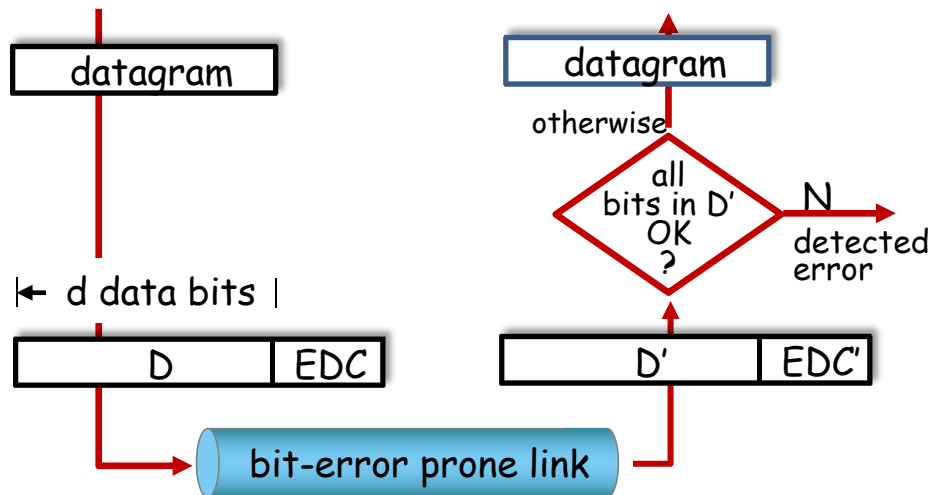




Error detection

EDC: error detection and correction bits (e.g., redundancy)

D: data protected by error checking, may include header fields



Error detection not 100% reliable!

- protocol may miss some errors, but rarely
- larger EDC field yields better detection and correction





Parity checking

single bit parity:

- detect single bit errors

0111000110101011	1
------------------	---

← d data bits →

parity bit

Even/odd parity: set parity bit so there is an even/odd number of 1's

At receiver:

- compute parity of d received bits
- compare with received parity bit - if different than error detected

Can detect **and** correct errors (without retransmission!)

- two-dimensional parity: detect **and correct** single bit errors

				row parity
	$d_{1,1}$...	$d_{1,j}$	$d_{1,j+1}$
	$d_{2,1}$...	$d_{2,j}$	$d_{2,j+1}$

	$d_{i,1}$...	$d_{i,j}$	$d_{i,j+1}$
column parity	$d_{i+1,1}$...	$d_{i+1,j}$	$d_{i+1,j+1}$

no errors:

1	0	1	0	1	1
1	1	1	1	0	0
0	1	1	1	0	1
0	0	1	0	1	0

detected and correctable single-bit error:

1	0	1	0	1	1
1	1	1	0	0	0
0	1	1	0	1	1
0	0	1	0	1	0

parity error





Internet checksum (review, see section 3.3)

Goal: detect errors (i.e., flipped bits) in transmitted segment

sender:

- treat contents of UDP segment (including UDP header fields and IP addresses) as sequence of 16-bit integers
- **checksum:** addition (one's complement sum) of segment content
- checksum value put into UDP checksum field

receiver:

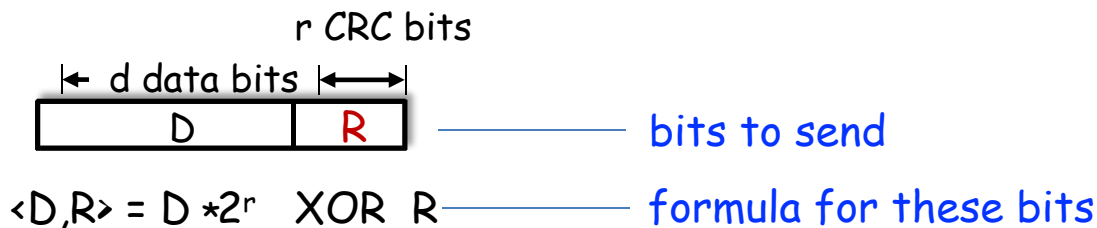
- compute checksum of received segment
- check if computed checksum equals checksum field value:
 - not equal - error detected
 - equal - no error detected. But maybe errors nonetheless? More later





Cyclic Redundancy Check (CRC)

- more powerful error-detection coding
- **D**: data bits (given, think of these as a binary number)
- **G**: bit pattern (generator), of $r+1$ bits (given, specified in CRC standard)



sender: compute r CRC bits, **R**, such that $\langle D, R \rangle$ exactly divisible by $G \pmod{2}$

- receiver knows G , divides $\langle D, R \rangle$ by G . If non-zero remainder: error detected!
- can detect all burst errors less than $r+1$ bits
- widely used in practice (Ethernet, 802.11 WiFi)





Cyclic Redundancy Check (CRC): example

Sender wants to compute R

such that:

$$D \cdot 2^r \text{ XOR } R = nG$$

... or equivalently (XOR R both sides):

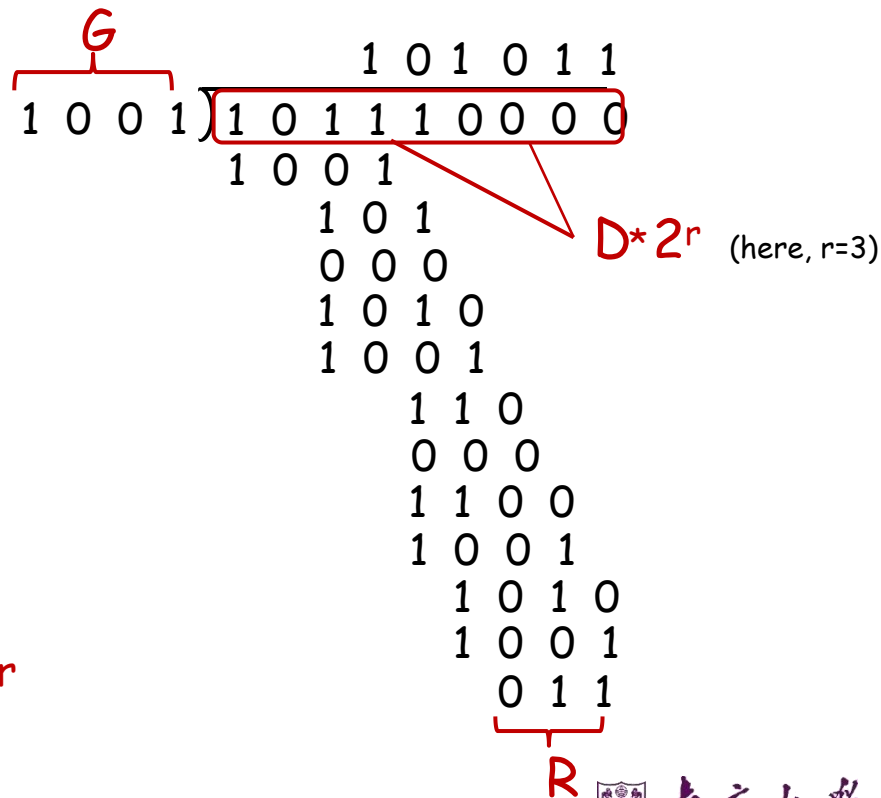
$$D \cdot 2^r = nG \text{ XOR } R$$

... which says:

if we divide $D \cdot 2^r$ by G , we
want remainder R to satisfy:

$$R = \text{remainder} \left[\frac{D \cdot 2^r}{G} \right]$$

algorithm for
computing R





Outline

- Introduction
- Error detection, correction
- Multiple access protocols
- LANs
- Link virtualization: MPLS
- Data center networking
- A day in the life of a web request





Multiple access links, protocols

two types of “links”:

- point-to-point
 - point-to-point link between Ethernet switch, host
 - PPP for dial-up access
- **broadcast (shared wire or medium)**
 - old-school Ethernet
 - upstream HFC in cable-based access network
 - 802.11 wireless LAN, 4G/4G. satellite



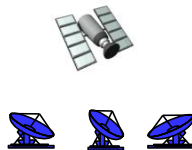
shared wire (e.g.,
cabled Ethernet)



shared radio: 4G/5G



shared radio: WiFi



shared radio: satellite



humans at a cocktail
party
(shared air, acoustical)



Multiple access protocols

- single shared broadcast channel
- two or more simultaneous transmissions by nodes: interference
 - **collision** if node receives two or more signals at the same time

multiple access protocol

- distributed algorithm that determines how nodes share channel, i.e., determine when node can transmit
- communication about channel sharing must use channel itself!
 - no out-of-band channel for coordination





An ideal multiple access protocol

given: multiple access channel (MAC) of rate R bps

desiderata:

1. when one node wants to transmit, it can send at rate R .
2. when M nodes want to transmit, each can send at average rate R/M
3. fully decentralized:
 - no special node to coordinate transmissions
 - no synchronization of clocks, slots
4. simple





MAC protocols: taxonomy

three broad classes:

- **channel partitioning**
 - divide channel into smaller “pieces” (time slots, frequency, code)
 - allocate piece to node for exclusive use
- **random access**
 - channel not divided, allow collisions
 - “recover” from collisions
- **“taking turns”**
 - nodes take turns, but nodes with more to send can take longer turns

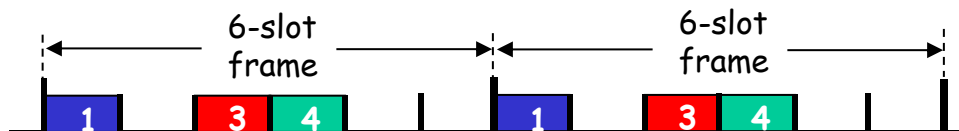




Channel partitioning MAC protocols: TDMA

TDMA: time division multiple access

- access to channel in "rounds"
- each station gets fixed length slot (length = packet transmission time) in each round
- unused slots go idle
- example: 6-station LAN, 1,3,4 have packets to send, slots 2,5,6 idle

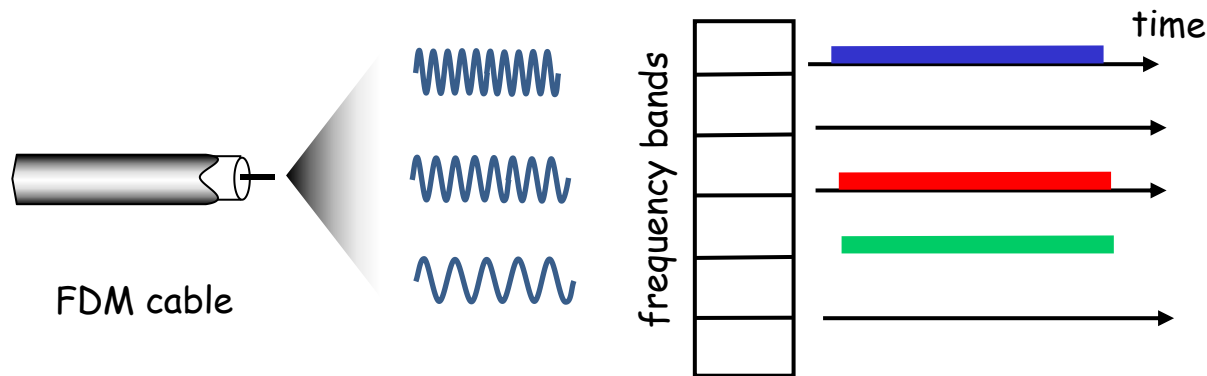




Channel partitioning MAC protocols: FDMA

FDMA: frequency division multiple access

- channel spectrum divided into frequency bands
- each station assigned fixed frequency band
- unused transmission time in frequency bands go idle
- example: 6-station LAN, 1,3,4 have packet to send, frequency bands 2,5,6 idle





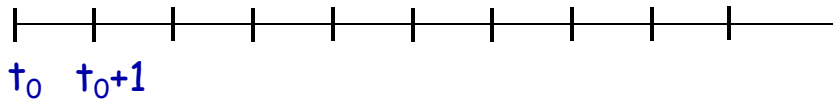
Random access protocols

- when node has packet to send
 - transmit at full channel data rate R
 - no a priori coordination among nodes
- two or more transmitting nodes: "collision"
- **random access protocol** specifies:
 - how to detect collisions
 - how to recover from collisions (e.g., via delayed retransmissions)
- examples of random access MAC protocols:
 - ALOHA, slotted ALOHA
 - CSMA, CSMA/CD, CSMA/CA





Slotted ALOHA



assumptions:

- all frames same size
- time divided into equal size slots (time to transmit 1 frame)
- nodes start to transmit only slot beginning
- nodes are synchronized
- if 2 or more nodes transmit in slot, all nodes detect collision

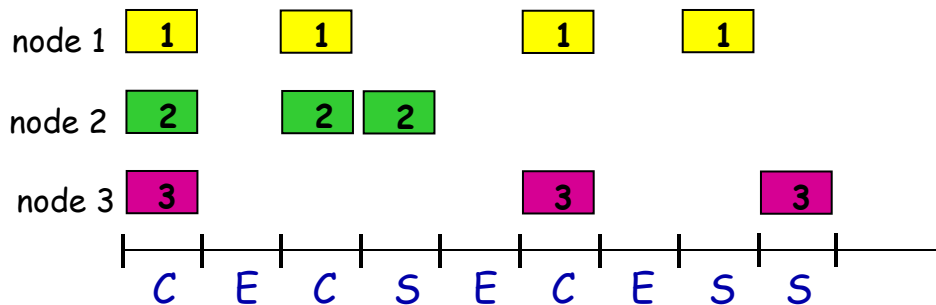
operation:

- when node obtains fresh frame, transmits in next slot
 - **if no collision:** node can send new frame in next slot
 - **if collision:** node retransmits frame in each subsequent slot with probability p until success

randomization - why?



Slotted ALOHA



C: collision
S: success
E: empty

Pros:

- single active node can continuously transmit at full rate of channel
- highly decentralized: only slots in nodes need to be in sync
- simple

Cons:

- collisions, wasting slots
- idle slots
- nodes may be able to detect collision in less than time to transmit packet
- clock synchronization





Slotted ALOHA: efficiency

efficiency: long-run fraction of successful slots (many nodes, all with many frames to send)

- suppose: N nodes with many frames to send, each transmits in slot with probability p
 - prob that given node has success in a slot = $p(1-p)^{N-1}$
 - prob that any node has a success = $Np(1-p)^{N-1}$
 - max efficiency: find p^* that maximizes $Np(1-p)^{N-1}$
 - for many nodes, take limit of $Np^*(1-p^*)^{N-1}$ as N goes to infinity, gives:

max efficiency = $1/e = .37$

- **at best:** channel used for useful transmissions 37% of time!





CSMA (carrier sense multiple access)

simple **CSMA**: listen before transmit:

- if channel sensed idle: transmit entire frame
- if channel sensed busy: defer transmission
- human analogy: don't interrupt others!

CSMA/CD: CSMA with collision detection

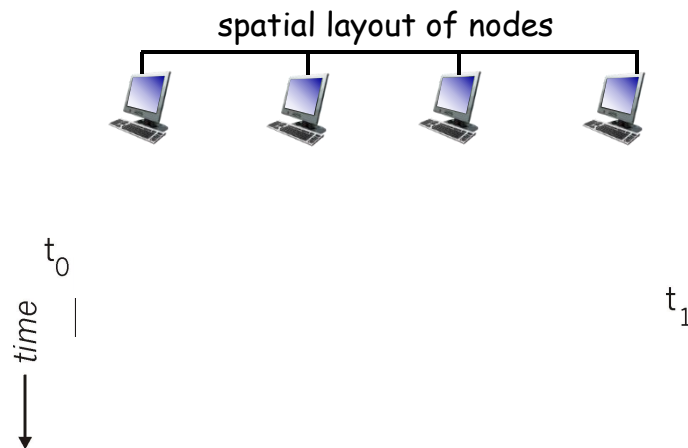
- collisions detected within short time
- colliding transmissions aborted, reducing channel wastage
- collision detection easy in wired, difficult with wireless
- human analogy: the polite conversationalist





CSMA: collisions

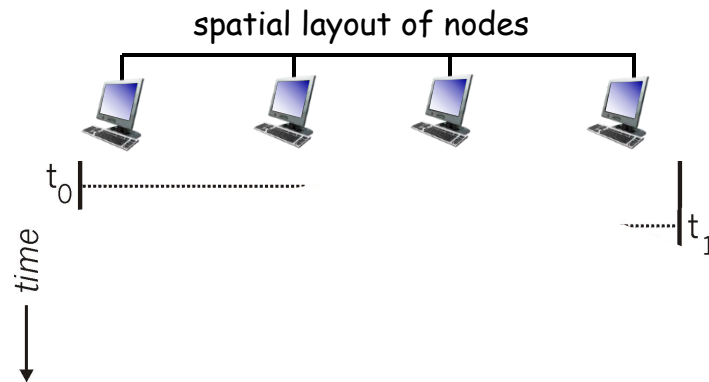
- collisions can still occur with carrier sensing:
 - **propagation delay** means two nodes may not hear each other's just-started transmission
- **collision**: entire packet transmission time wasted
 - distance & propagation delay play role in determining collision probability





CSMA/CD:

- CSMA/CD reduces the amount of time wasted in collisions
 - transmission aborted on collision detection





Ethernet CSMA/CD algorithm

1. Ethernet receives datagram from network layer, creates frame
2. If Ethernet senses channel:
 - if **idle**: start frame transmission.
 - if **busy**: wait until channel idle, then transmit
3. If entire frame transmitted without collision - done!
4. If another transmission detected while sending: abort, send jam signal
5. After aborting, enter **binary (exponential) backoff**:
 - after m th collision, chooses K at random from $\{0, 1, 2, \dots, 2^m - 1\}$. Ethernet waits $K \cdot 512$ bit times, returns to Step 2
 - more collisions: longer backoff interval





"Taking turns" MAC protocols

channel partitioning MAC protocols:

- share channel efficiently and fairly at high load
- inefficient at low load: delay in channel access, $1/N$ bandwidth allocated even if only 1 active node!

random access MAC protocols

- efficient at low load: single node can fully utilize channel
- high load: collision overhead

"taking turns" protocols

- look for best of both worlds!

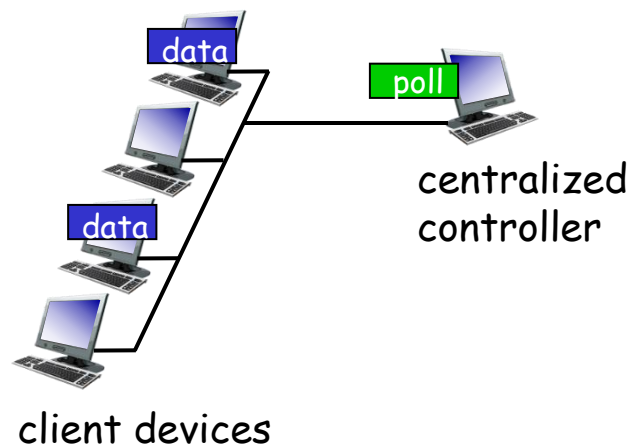




"Taking turns" MAC protocols

polling:

- centralized controller "invites" other nodes to transmit in turn
- typically used with "dumb" devices
- concerns:
 - polling overhead
 - latency
 - single point of failure (master)
- Bluetooth uses polling

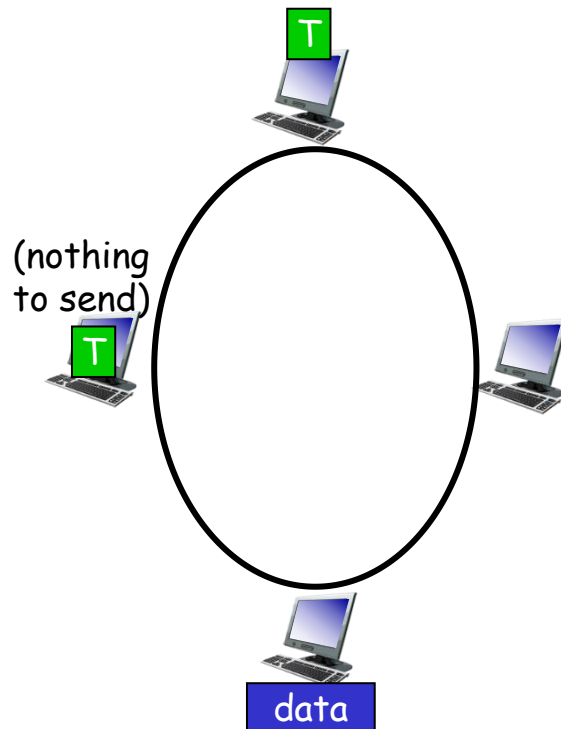




"Taking turns" MAC protocols

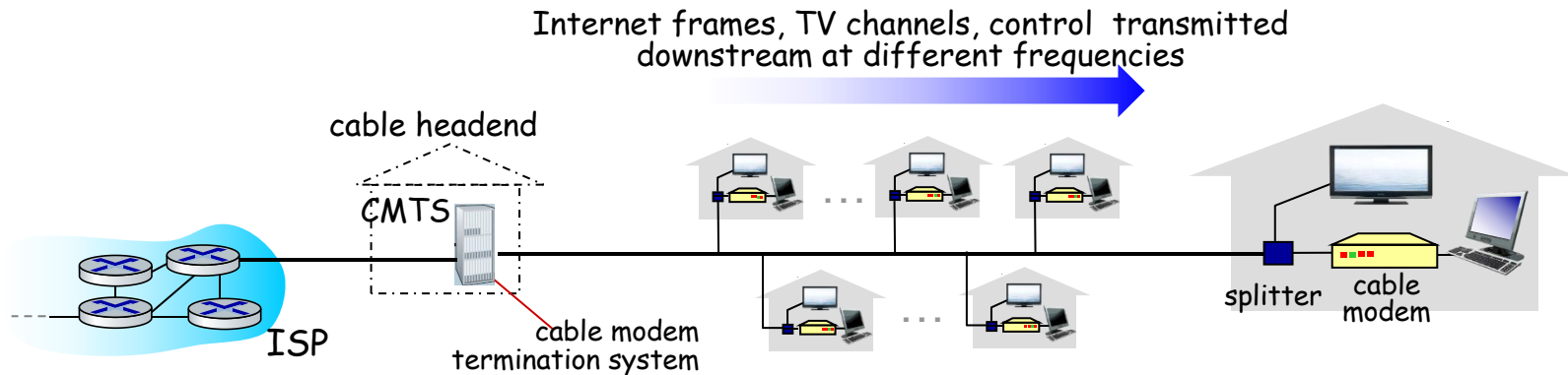
token passing:

- control **token** message explicitly passed from one node to next, sequentially
 - transmit while holding token
- concerns:
 - token overhead
 - latency
 - single point of failure (token)





Cable access network: FDM, TDM and random access!

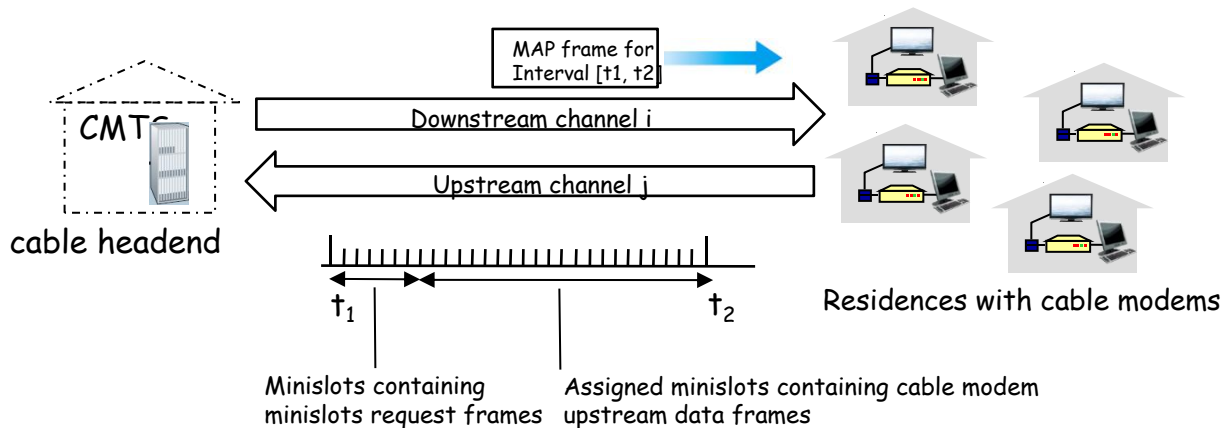


- **multiple** downstream (broadcast) FDM channels: up to 1.6 Gbps/channel
 - single CMTS transmits into channels
- **multiple** upstream channels (up to 1 Gbps/channel)
 - **multiple access**: all users contend (random access) for certain upstream channel time slots; others assigned TDM





Cable access network



DOCSIS: data over cable service interface specification

- FDM over upstream, downstream frequency channels
- TDM upstream: some slots assigned, some have contention
 - downstream MAP frame: assigns upstream slots
 - request for upstream slots (and data) transmitted random access (binary backoff) in selected slots





Summary of MAC protocols

- **channel partitioning**, by time, frequency or code
 - Time Division, Frequency Division
- **random access** (dynamic),
 - ALOHA, S-ALOHA, CSMA, CSMA/CD
 - carrier sensing: easy in some technologies (wire), hard in others (wireless)
 - CSMA/CD used in Ethernet
 - CSMA/CA used in 802.11
- **taking turns**
 - polling from central site, token passing
 - Bluetooth, FDDI, token ring





Outline

- Introduction
- Error detection, correction
- Multiple access protocols
- LANs
- Link virtualization: MPLS
- Data center networking
- A day in the life of a web request





MAC addresses

- 32-bit IP address:
 - network-layer address for interface
 - used for layer 3 (network layer) forwarding
 - e.g.: 128.119.40.136
- MAC (or LAN or physical or Ethernet) address:
 - function: used "locally" to get frame from one interface to another physically-connected interface (same subnet, in IP-addressing sense)
 - 48-bit MAC address (for most LANs) burned in NIC ROM, also sometimes software settable
 - e.g.: 1A-2F-BB-76-09-AD
 - hexadecimal (base 16) notation
(each "numeral" represents 4 bits)

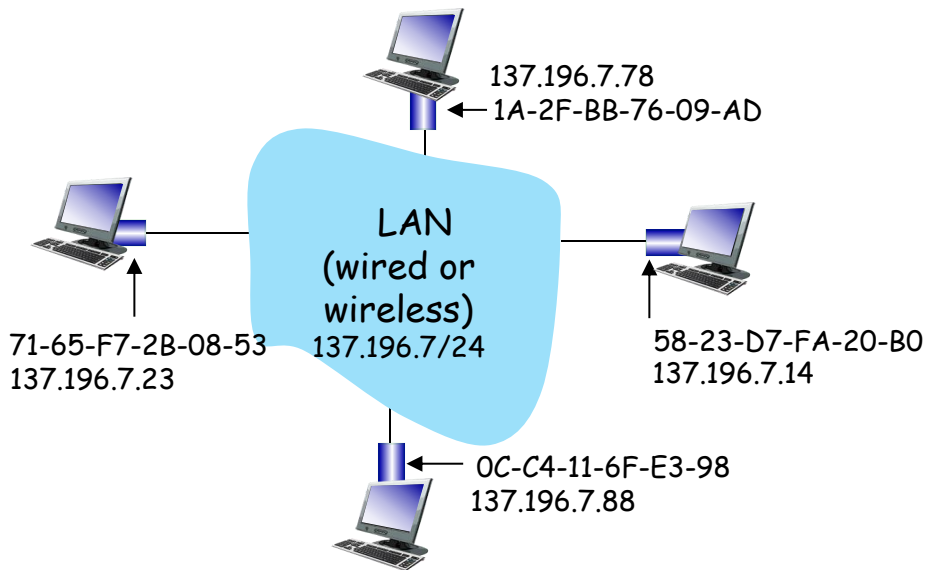




MAC addresses

each interface on LAN

- has unique 48-bit **MAC** address
- has a locally unique 32-bit IP address (as we've seen)





MAC addresses

- MAC address allocation administered by IEEE
- manufacturer buys portion of MAC address space (to assure uniqueness)
- analogy:
 - MAC address: like Social Security Number
 - IP address: like postal address
- MAC flat address: portability
 - can move interface from one LAN to another
 - recall IP address *not* portable: depends on IP subnet to which node is attached

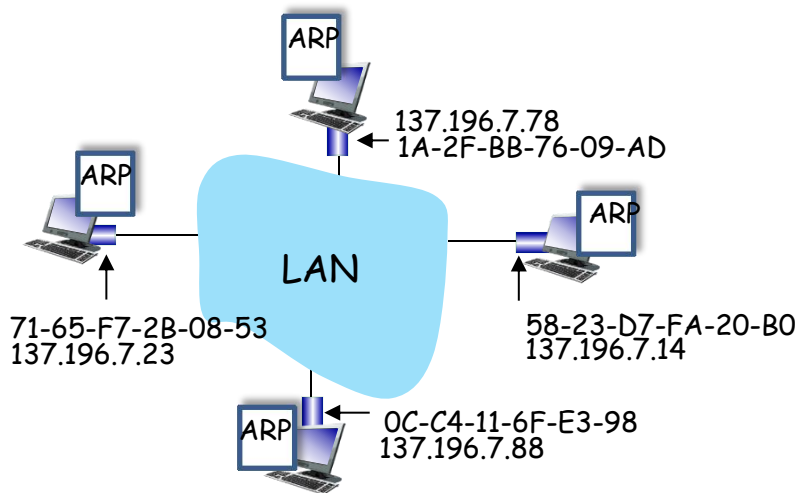




ARP: address resolution protocol

Question: how to determine interface's MAC address, knowing its IP address?

ARP table: each IP node (host, router) on LAN has table



- IP/MAC address mappings for some LAN nodes:

< IP address; MAC address; TTL >

- TTL (Time To Live): time after which address mapping will be forgotten (typically 20 min)





ARP protocol in action

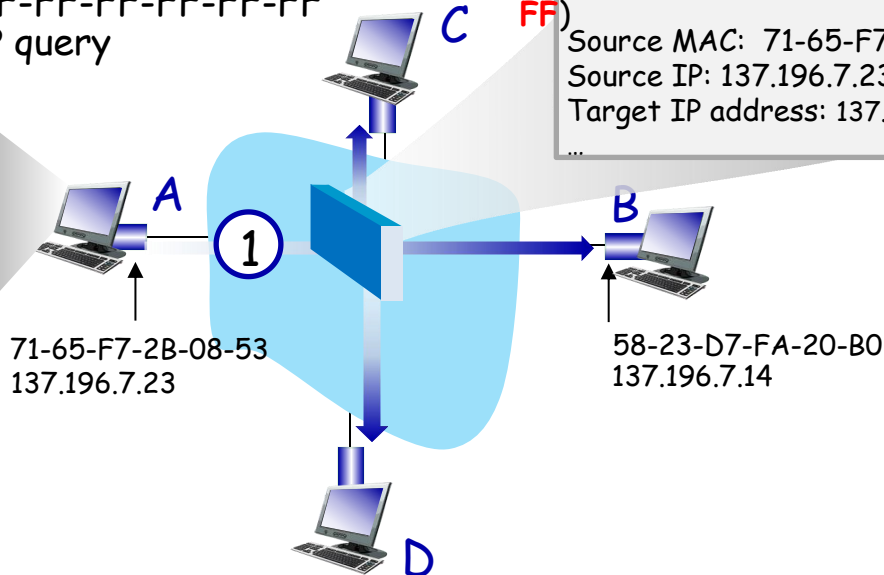
example: A wants to send datagram to B

- B's MAC address not in A's ARP table, so A uses ARP to find B's MAC address

- ① A broadcasts ARP query, containing B's IP addr
- destination MAC address = FF-FF-FF-FF-FF-FF
 - all nodes on LAN receive ARP query

ARP table in A

IP addr	MAC addr	TTL



Ethernet frame (sent to **FF-FF-FF-FF-FF-FF**)

Source MAC: 71-65-F7-2B-08-53
Source IP: 137.196.7.23
Target IP address: 137.196.7.14

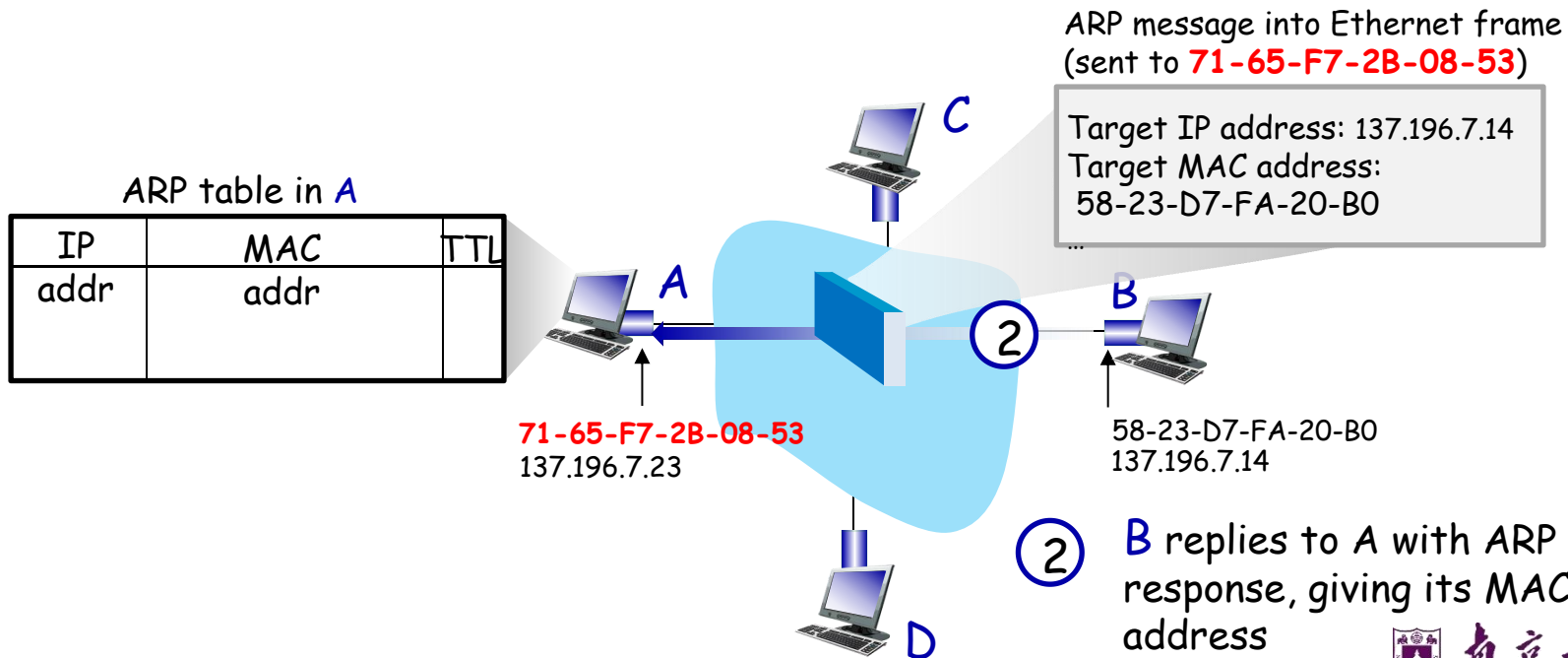




ARP protocol in action

example: A wants to send datagram to B

- B's MAC address not in A's ARP table, so A uses ARP to find B's MAC address

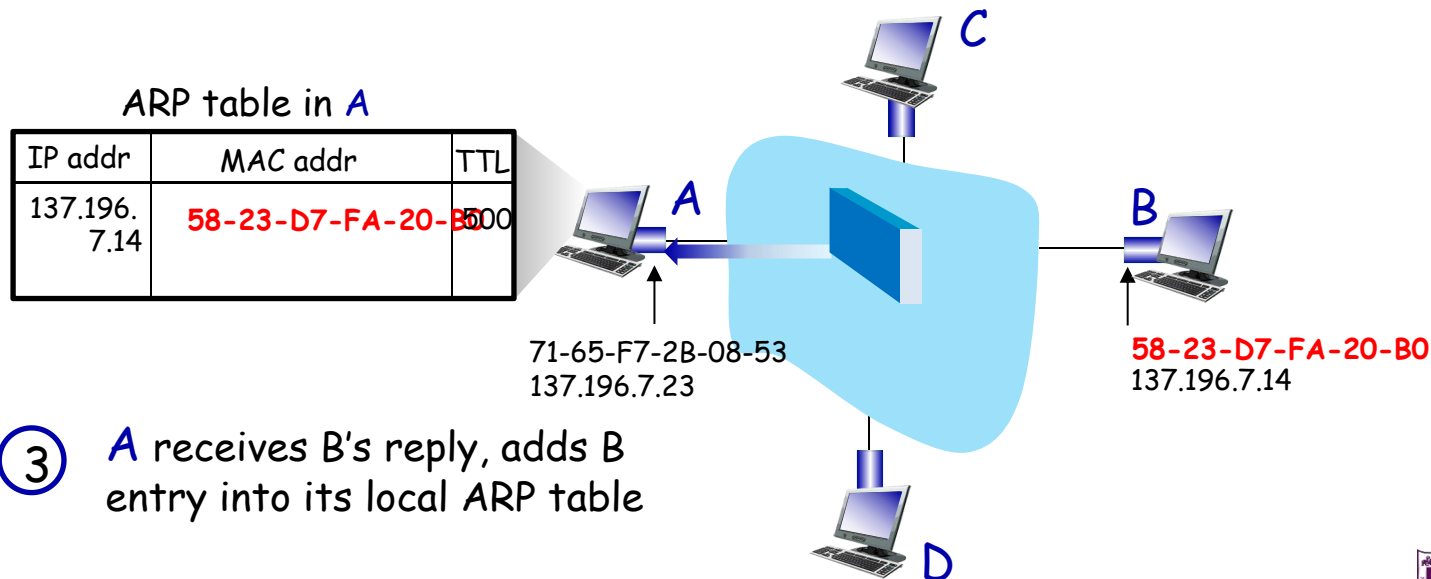




ARP protocol in action

example: A wants to send datagram to B

- B's MAC address not in A's ARP table, so A uses ARP to find B's MAC address

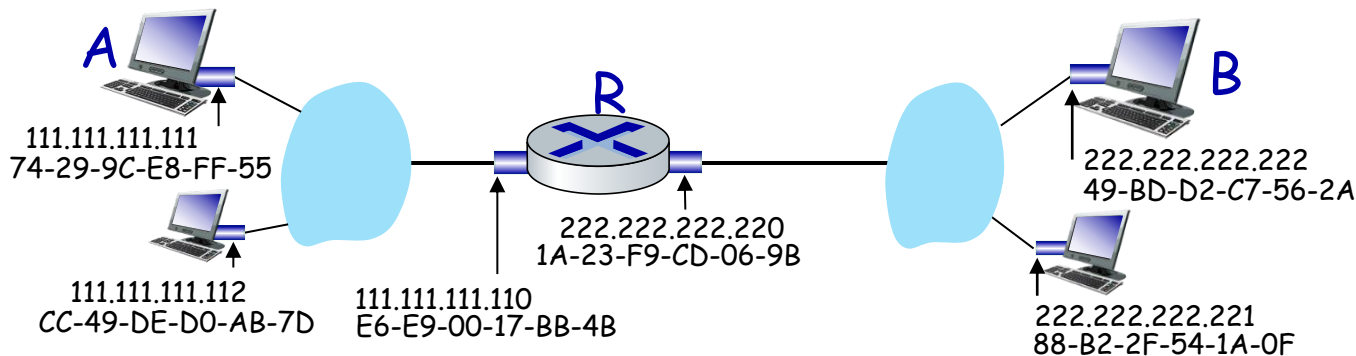




Routing to another subnet: addressing

walkthrough: sending a datagram from A to B via R

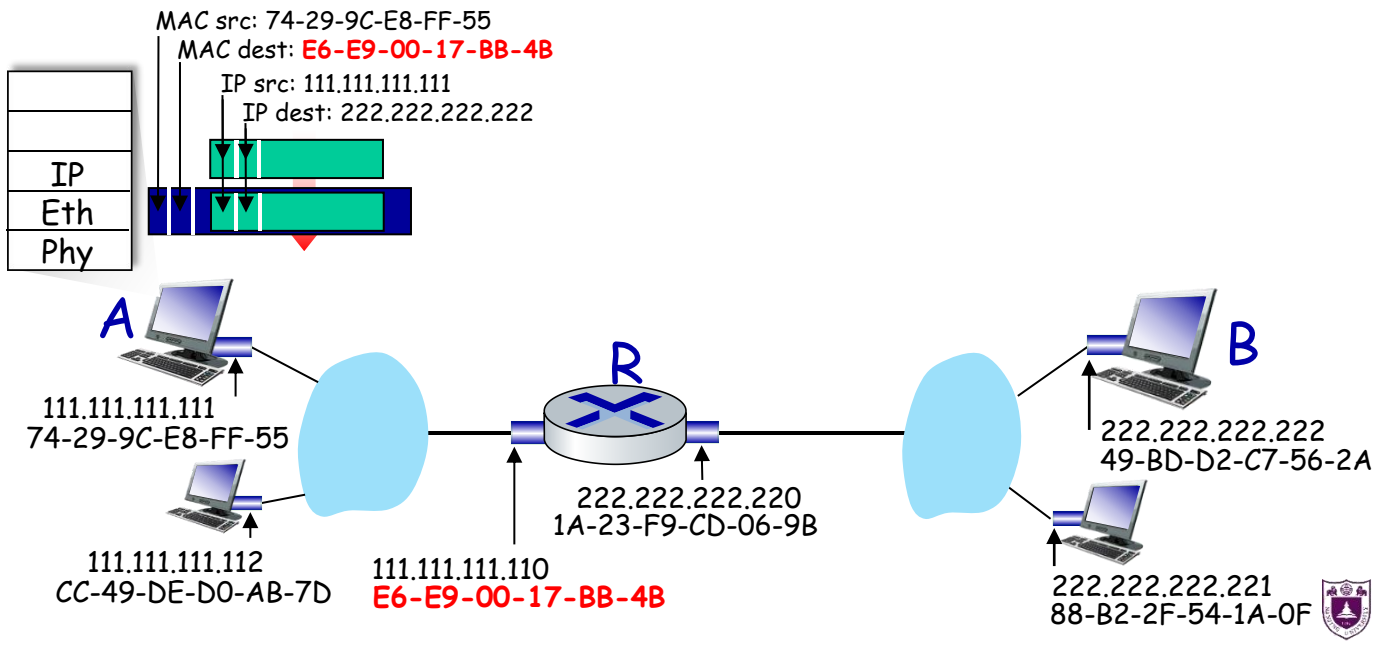
- focus on addressing - at IP (datagram) and MAC layer (frame) levels
- assume that:
 - A knows B's IP address
 - A knows IP address of first hop router, R (how?)
 - A knows R's MAC address (how?)





Routing to another subnet: addressing

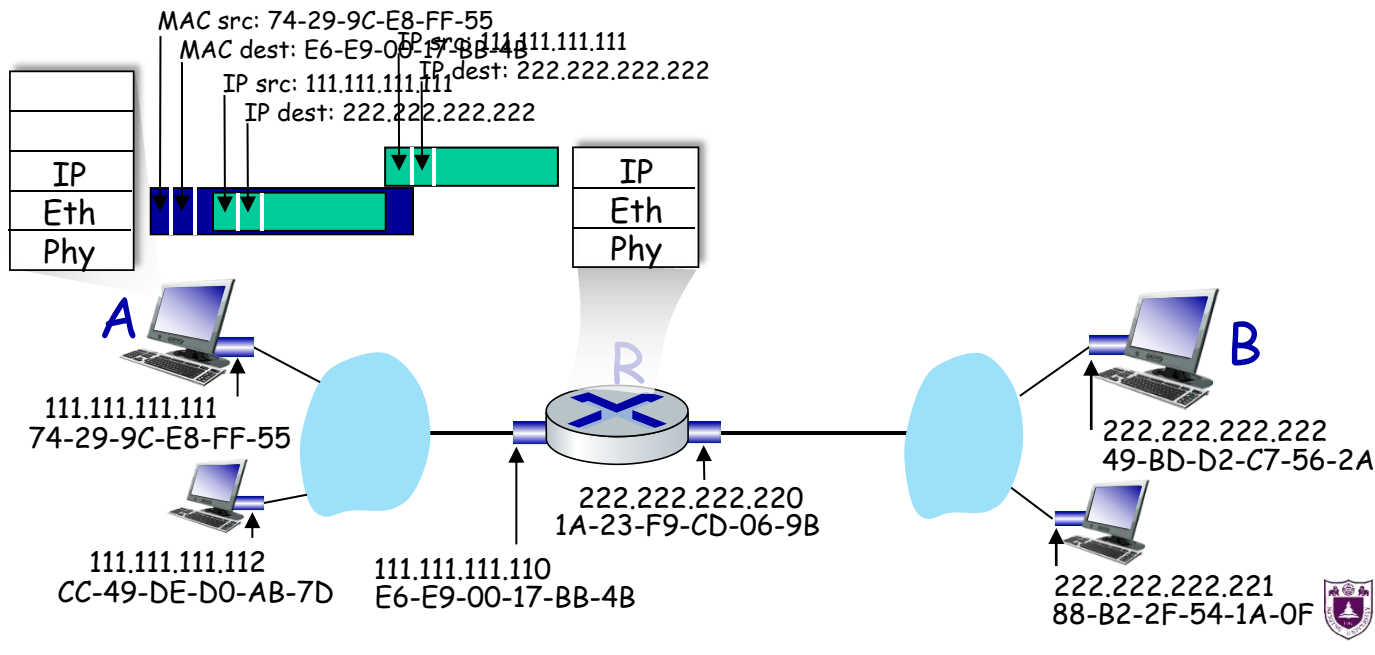
- A creates IP datagram with IP source A, destination B
- A creates link-layer frame containing A-to-B IP datagram
- R's MAC address is frame's destination





Routing to another subnet: addressing

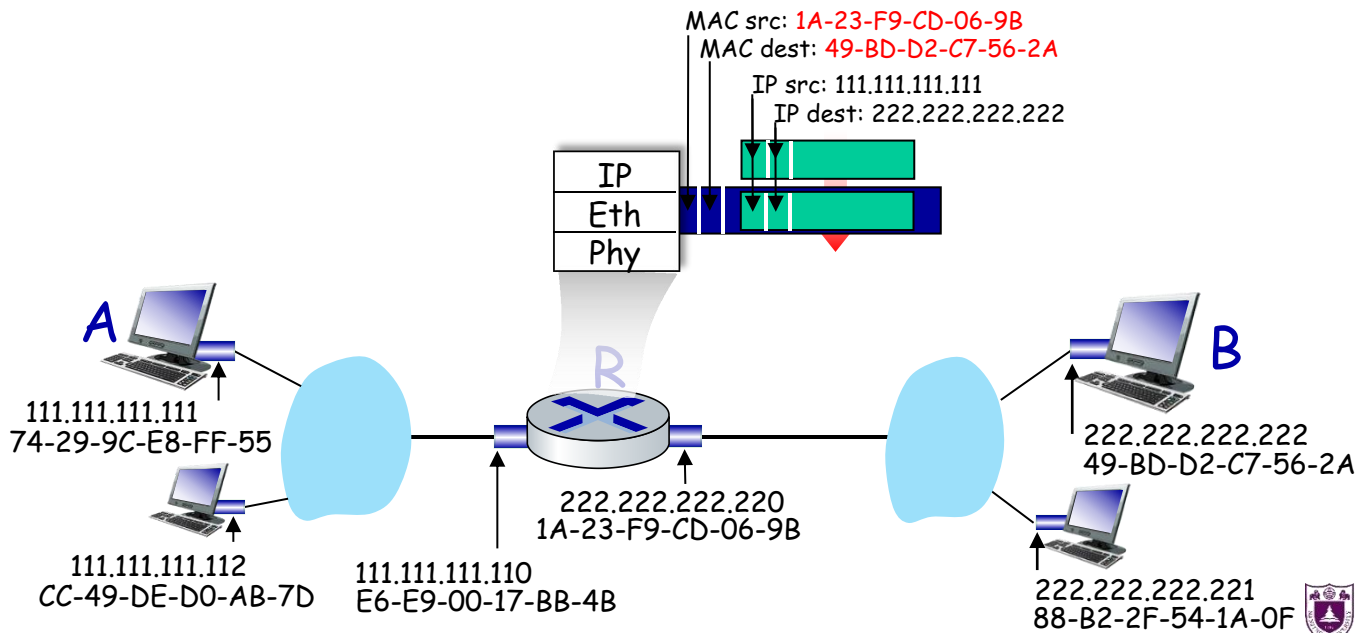
- frame sent from A to R
- frame received at R, datagram removed, passed up to IP





Routing to another subnet: addressing

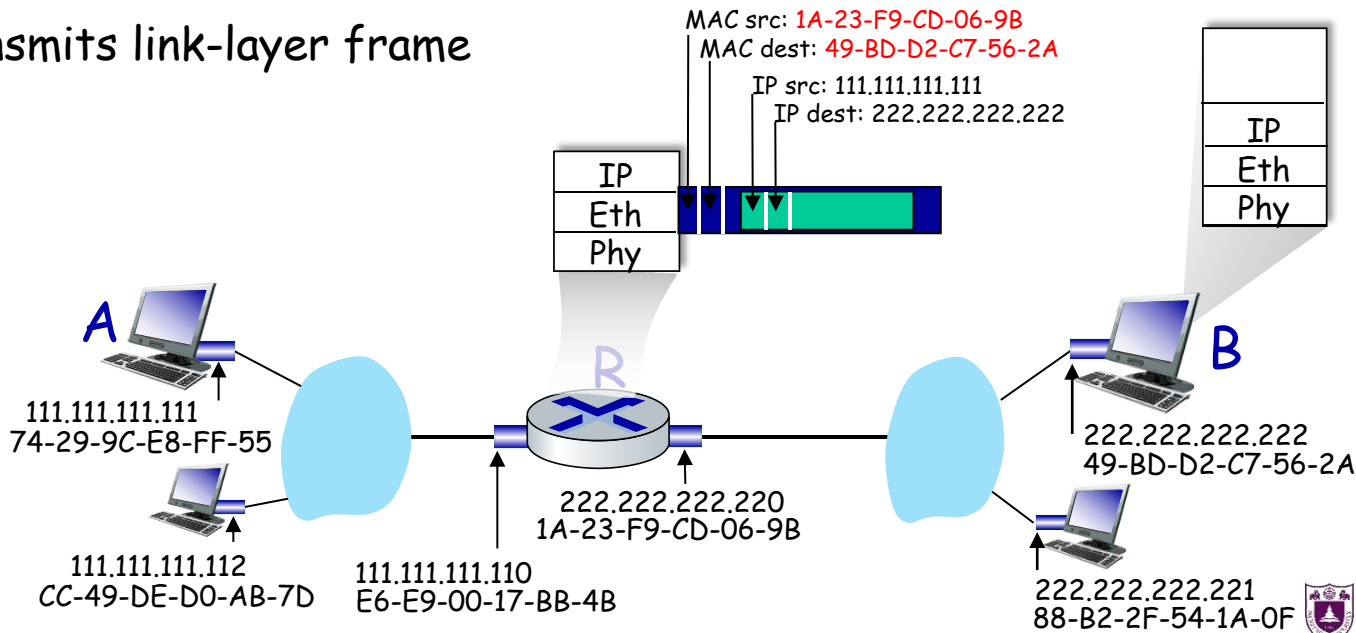
- R determines outgoing interface, passes datagram with IP source A, destination B to link layer
- R creates link-layer frame containing A-to-B IP datagram. Frame destination address: B's MAC address





Routing to another subnet: addressing

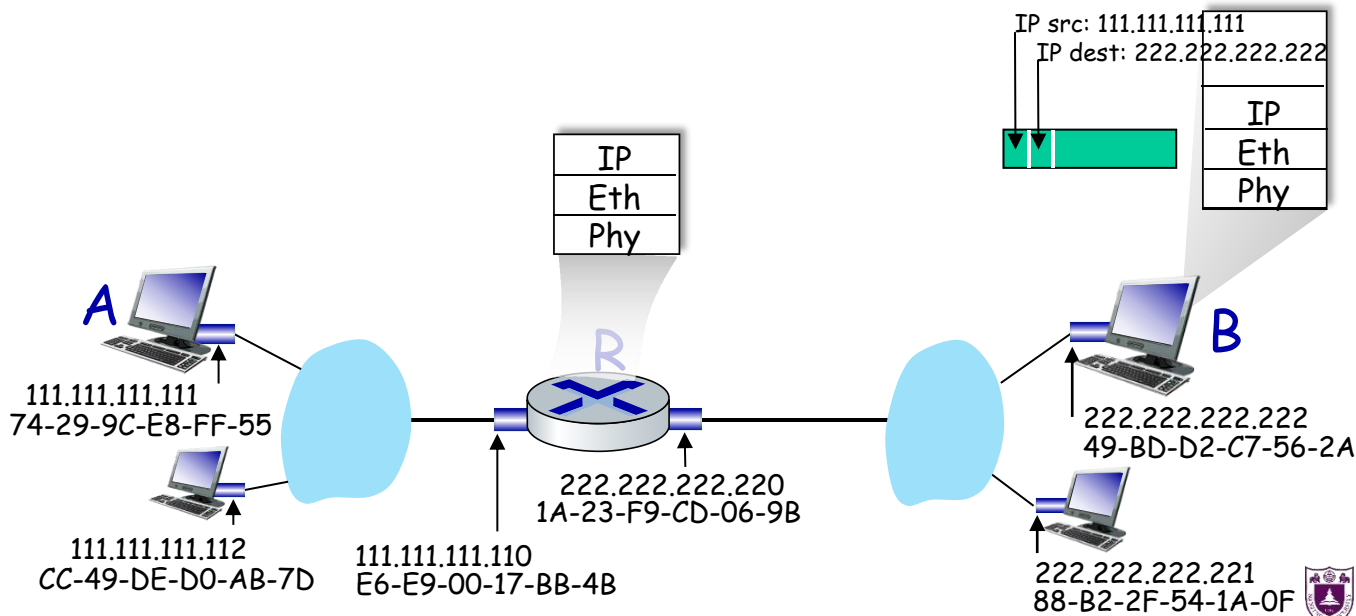
- R determines outgoing interface, passes datagram with IP source A, destination B to link layer
- R creates link-layer frame containing A-to-B IP datagram. Frame destination address: B's MAC address
- transmits link-layer frame





Routing to another subnet: addressing

- B receives frame, extracts IP datagram destination B
- B passes datagram up protocol stack to IP





提问

Q & A



南京大學
NANJING UNIVERSITY