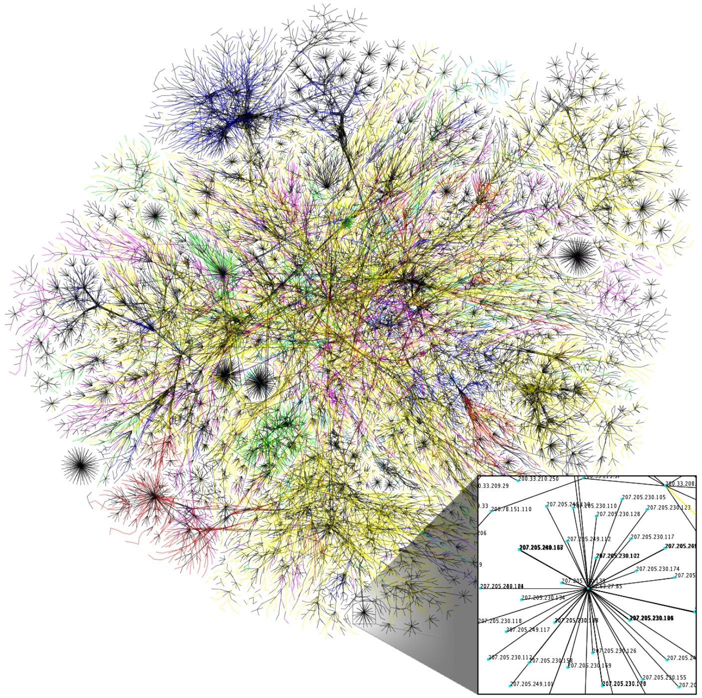
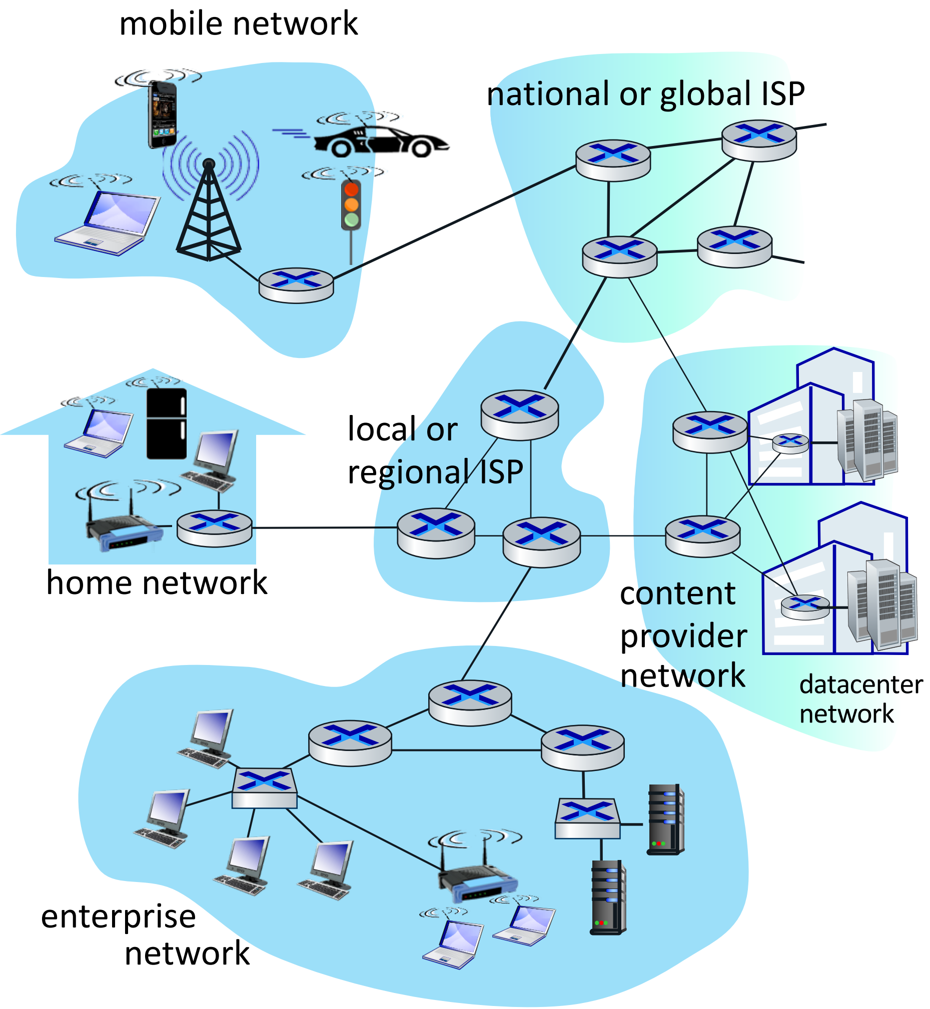
Fundamental Concepts

- Two key functions:

  Routing and Forwarding

- Queueing?



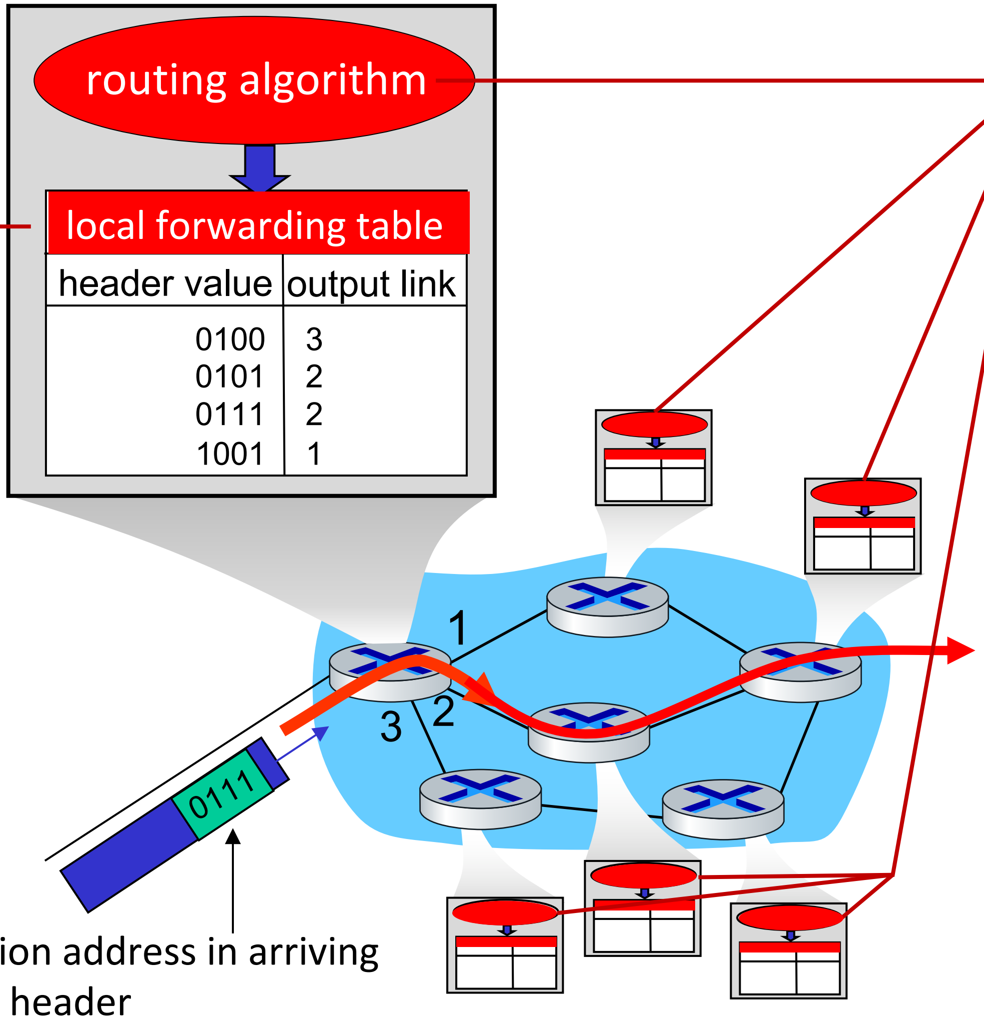


Two key functions: Routing and Forwarding

Forwarding:

aka “switching”

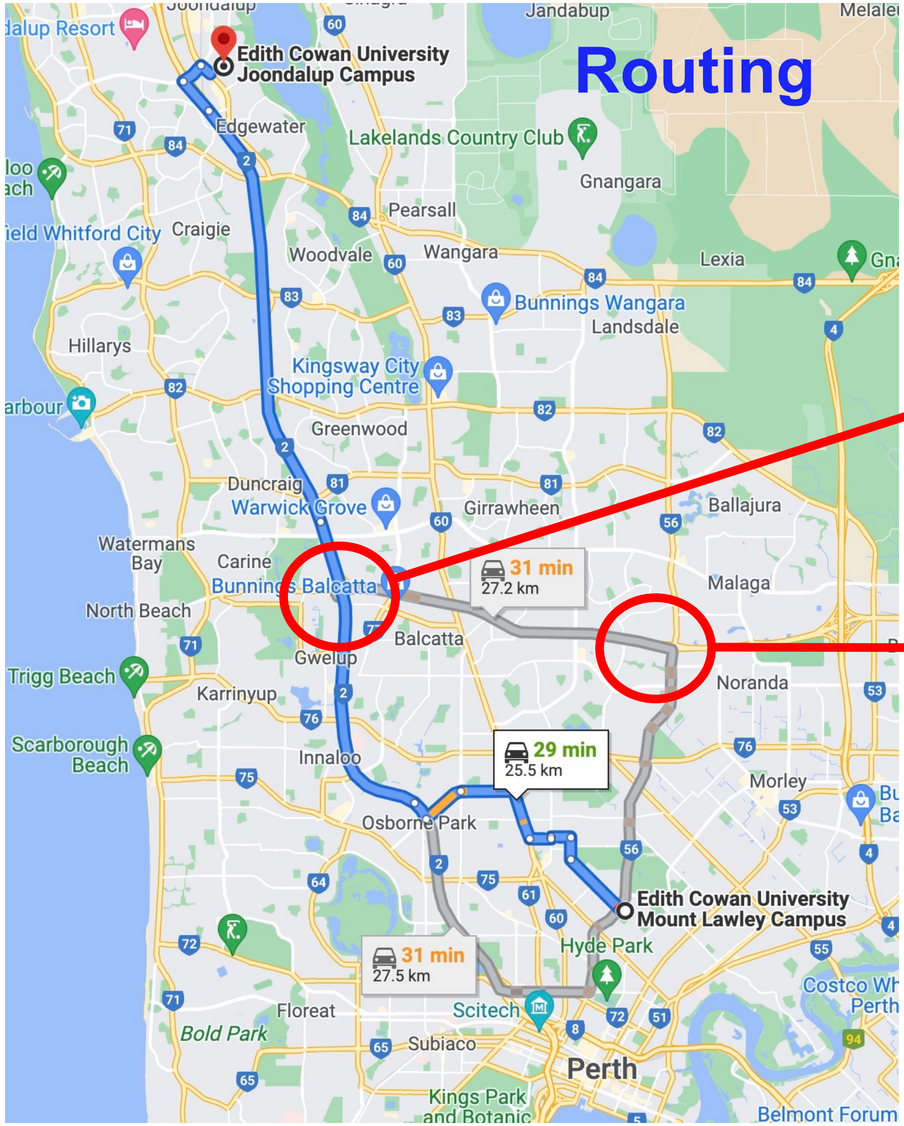
local action: move arriving packets from router’s input link to appropriate router output link

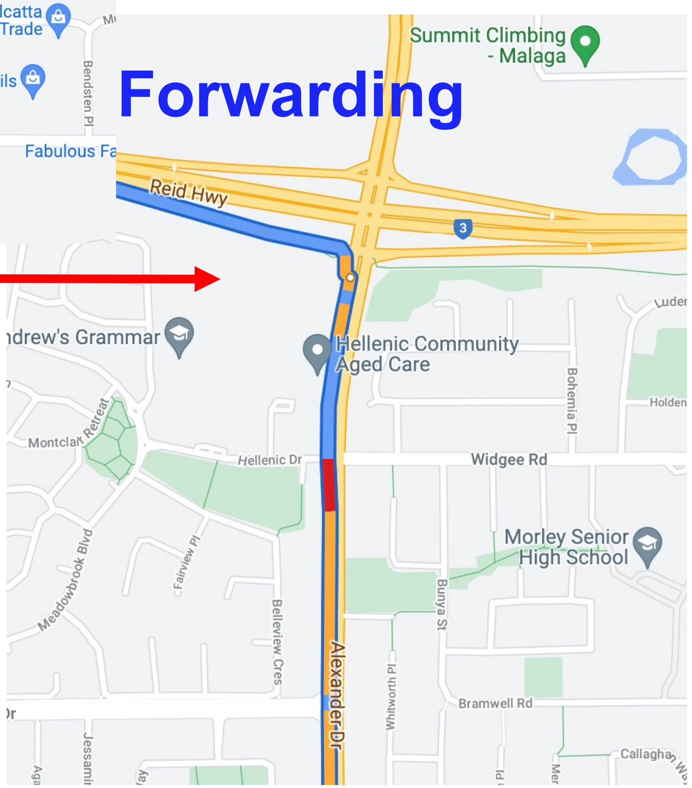


Routing:

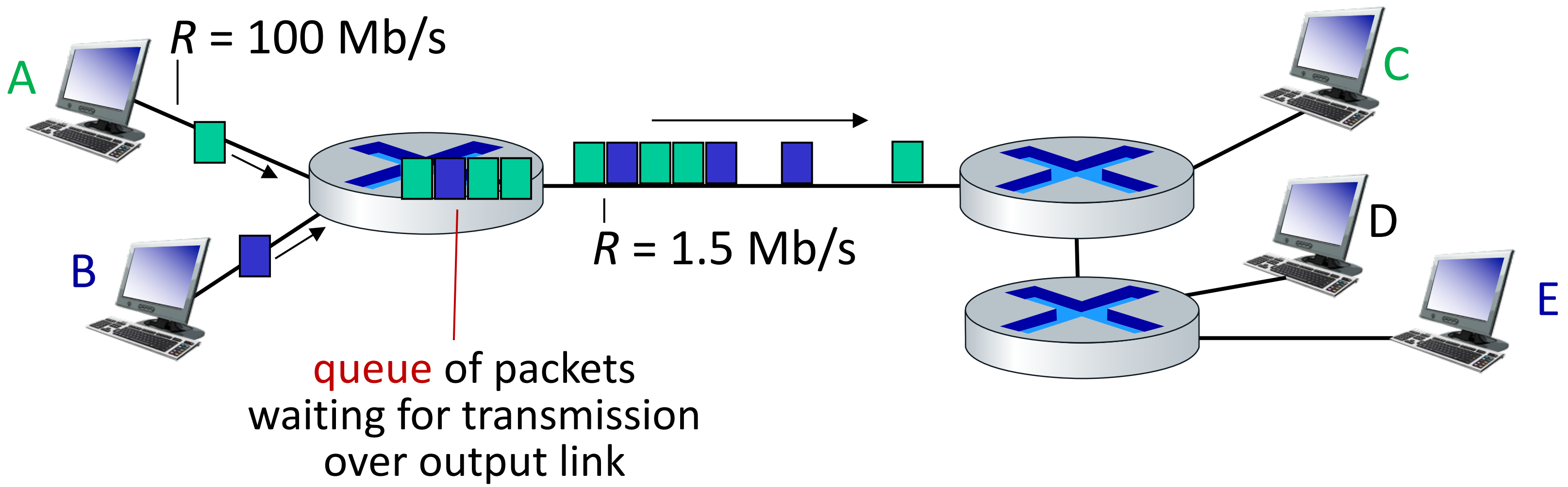
- *global* action: determine source-destination paths taken by packets
- routing algorithms

Two key functions: Routing and Forwarding



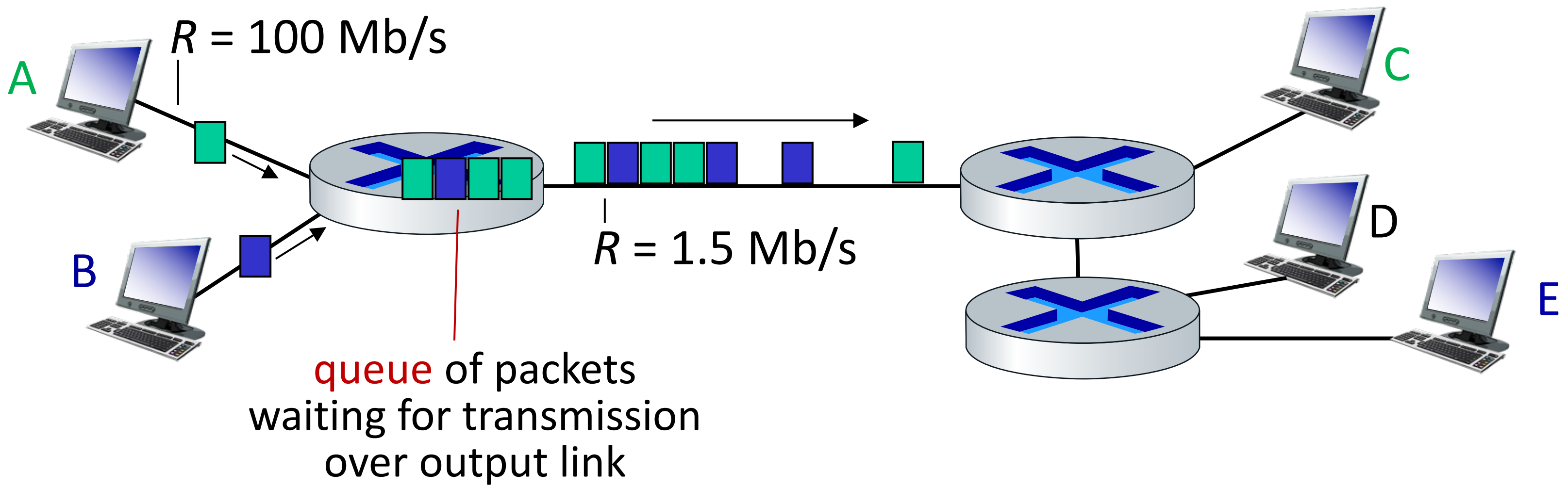


Packet-switching: Queueing



Queueing occurs when work arrives faster than it can be serviced:

Packet-switching: Queueing



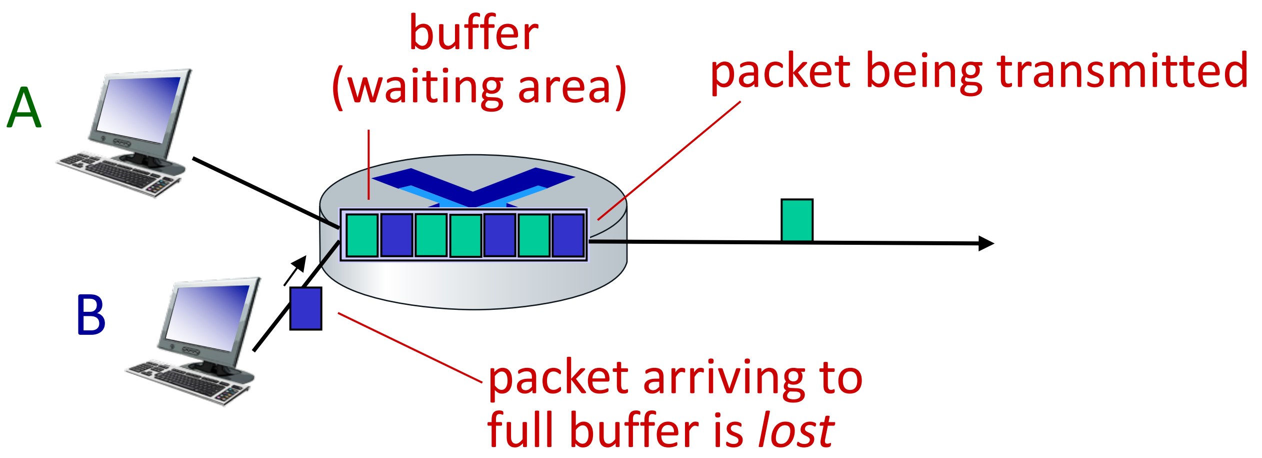
Packet queuing and loss: if arrival rate (in bps) to link exceeds transmission rate (bps) of link for some period of time:

packets will queue, waiting to be transmitted on output link

packets can be dropped (lost) if memory (buffer) in router fills up

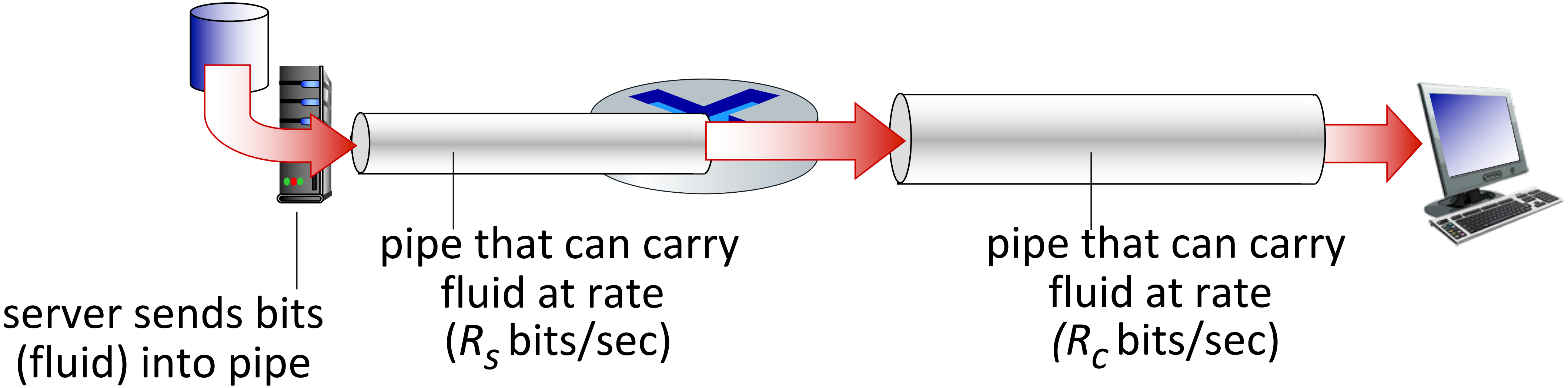
Packet Loss

- queue (aka buffer) preceding link in buffer has finite capacity
- packet arriving to full queue dropped (aka lost)
- lost packet may be retransmitted by previous node, by source end system, or not at all



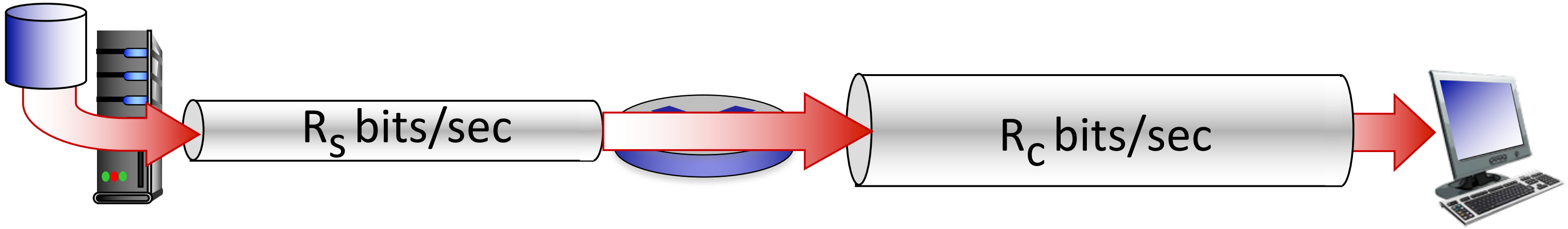
Throughput

- *throughput:* rate (bits/time unit) at which bits are being sent from sender to receiver
  - *instantaneous:* rate at given point in time
  - *average:* rate over longer period of time

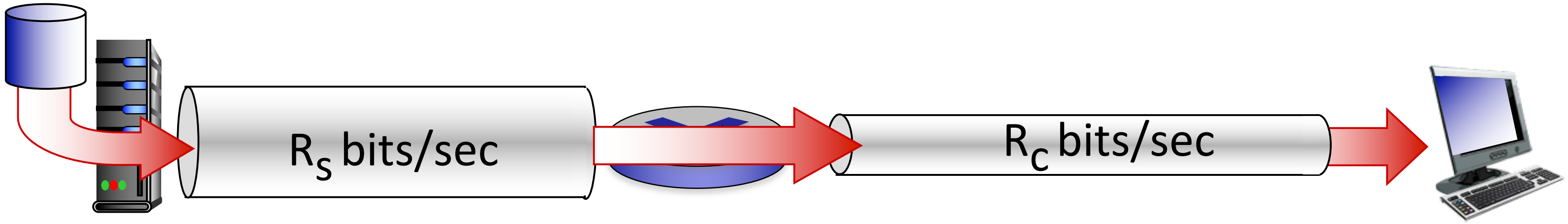


Throughput

$R_s < R_c$ What is average end-end throughput?



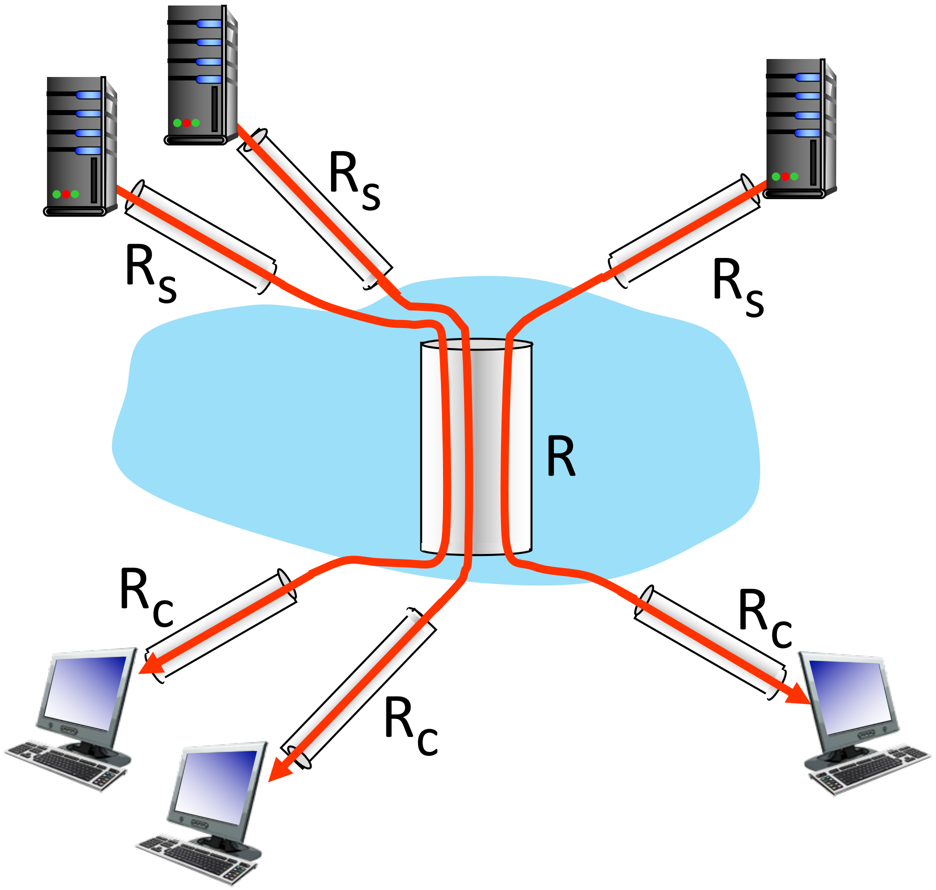
$R_s > R_c$ What is average end-end throughput?



bottleneck link

link on end-end path that constrains end-end throughput

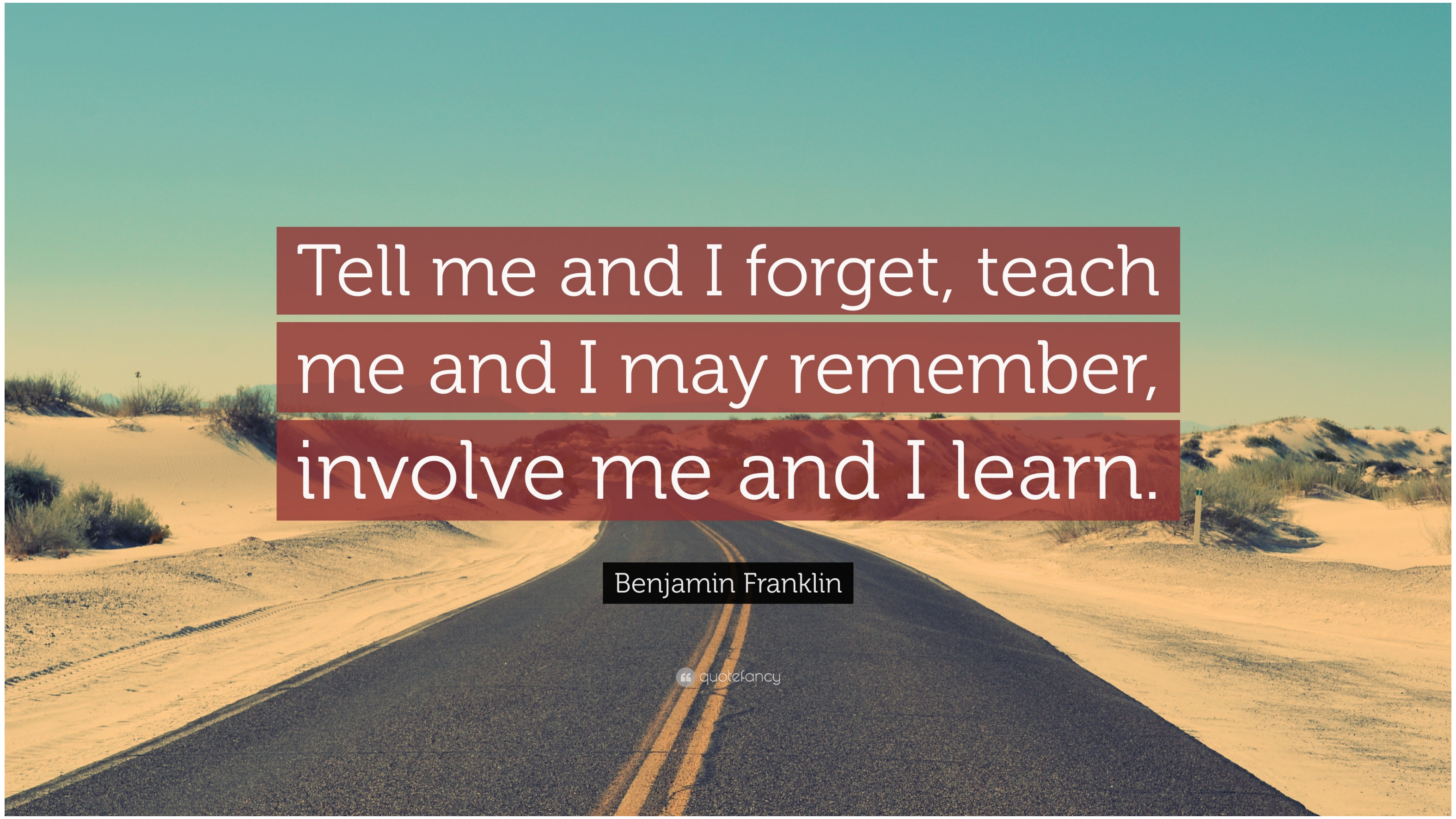
Throughput: Where is the bottleneck?



10 connections (fairly) share backbone bottleneck link *R* bits/sec

- per-connection end-end throughput: *min*(R_c,R_s,$R/10$)
- in practice: R_c or R_s is often bottleneck

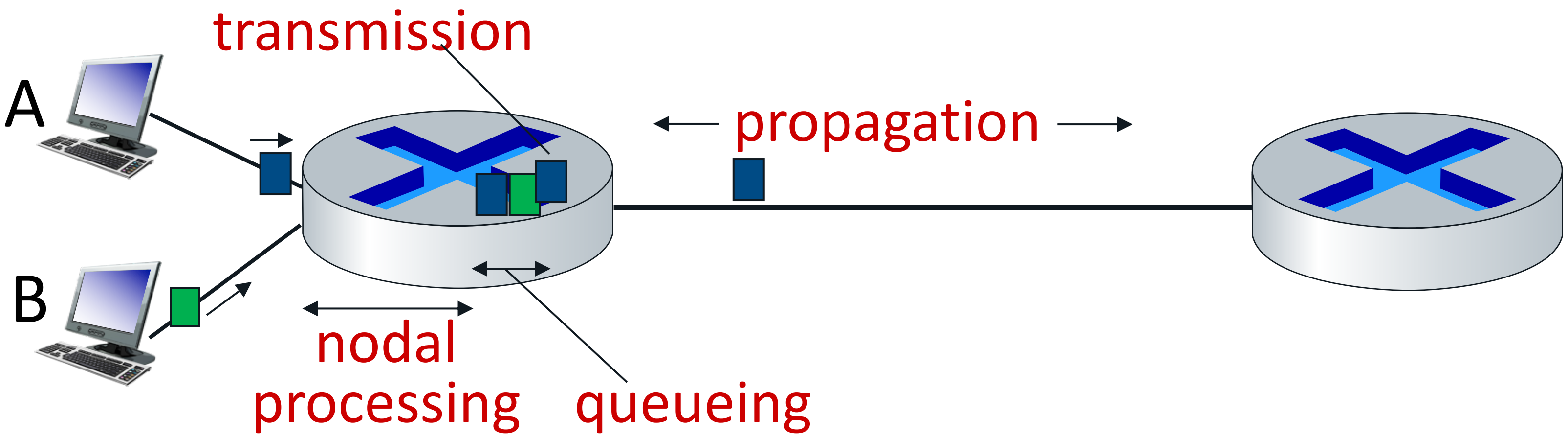
Thank You!



Tell me and I forget, teach
me and I may remember,
involve me and I learn.
Benjamin Franklin
quotefancy

Network Delay

Packet delay: Four Sources

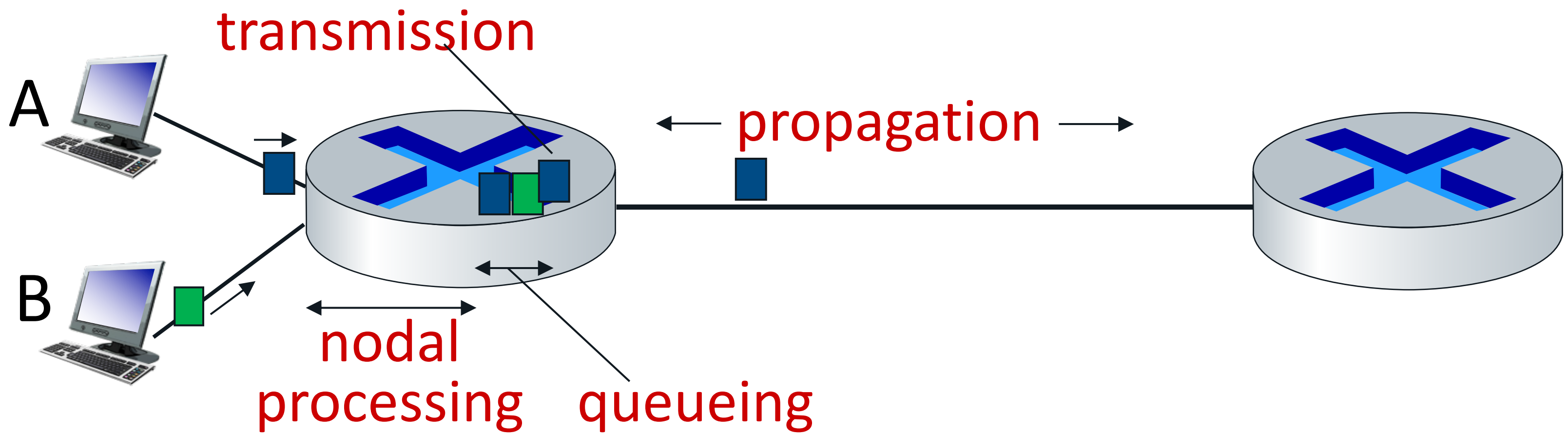


$$d_{nodal} = d_{proc} + d_{queue} + d_{trans} + d_{prop}$$

d_{proc}: nodal processing
- check bit errors
- determine output link
- typically < microsecs

d_{queue}: queueing delay
- time waiting at output link for transmission
- depends on congestion level of router

Packet delay: Four Sources

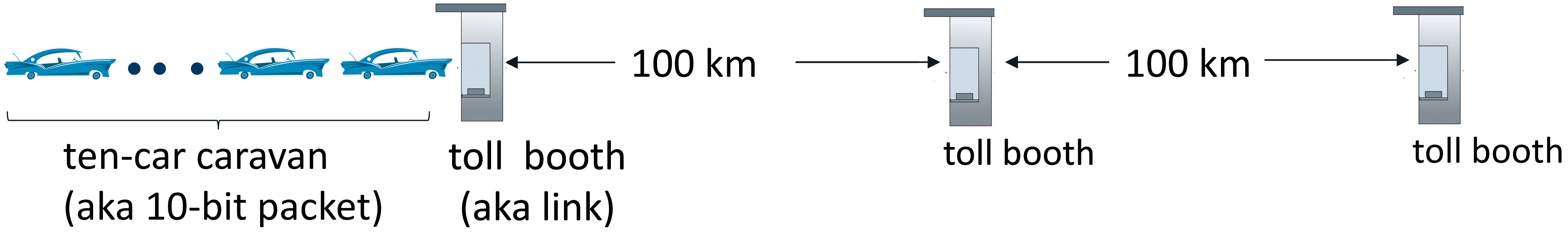


$$d_{nodal} = d_{proc} + d_{queue} + d_{trans} + d_{prop}$$

d_{prop}: propagation delay:

- d: length of physical link
- s: propagation speed (~2x10^8 m/sec)
- $d_{prop} = d/s$

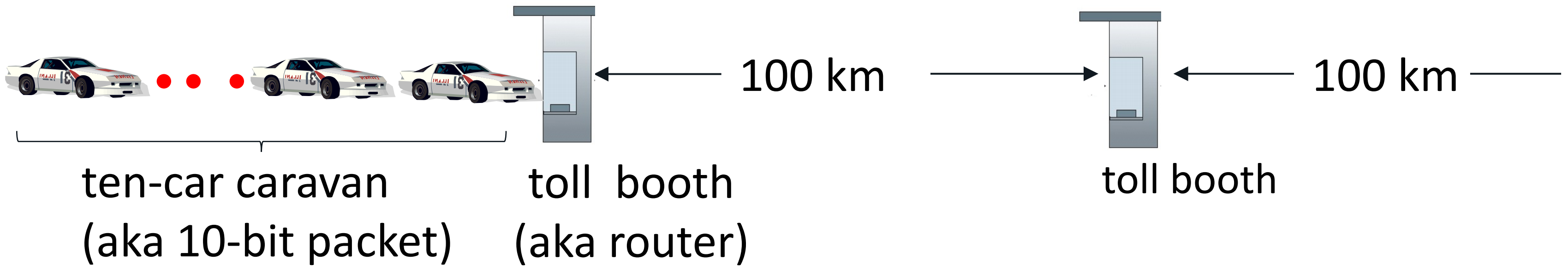
Car Analogy



- car ~ bit; caravan ~ packet; toll service ~ link transmission
- toll booth takes 12 sec to service car (bit transmission time)
- "propagate" at 100 km/hr
- *Q:* How long until caravan is lined up before 2nd toll booth?

- time to "push" entire caravan through toll booth onto highway = 12*10 = 120 sec
- time for last car to propagate from 1st to 2nd toll both: 100km/(100km/hr) = 1 hr
- *A:* 62 minutes

Car Analogy



- suppose cars now "propagate" at 1000 km/hr
- and suppose toll booth now takes one min to service a car
- *Q: Will cars arrive to 2nd booth before all cars serviced at first booth?*

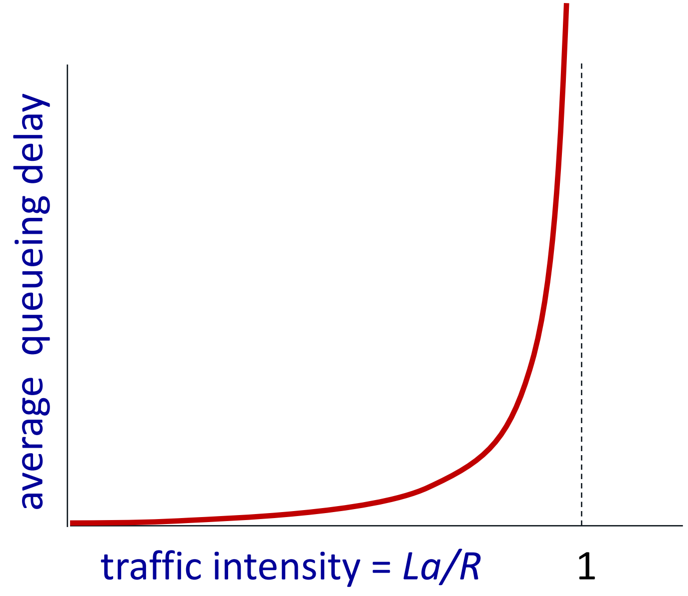
A: Yes! after 7 min, first car arrives at second booth; three cars still at first booth

Packet Queueing Delay

- *a:* average packet arrival rate
- *L:* packet length (bits)
- *R:* link bandwidth (bit transmission rate)

$$\frac{L \cdot a}{R} : \quad \frac{\text{arrival rate of bits}}{\text{service rate of bits}}$$

"traffic intensity"



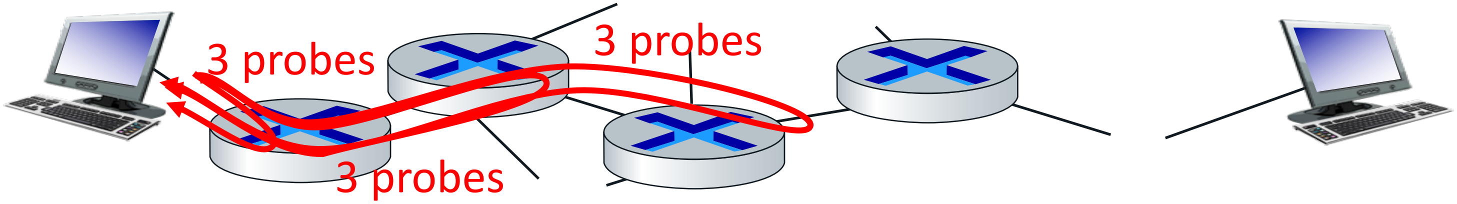
- *La/R* ~ 0: avg. queueing delay small
- *La/R* -> 1: avg. queueing delay large
- *La/R* > 1: more "work" arriving is more than can be serviced - average delay infinite!

La/R ~ 0

La/R -> 1

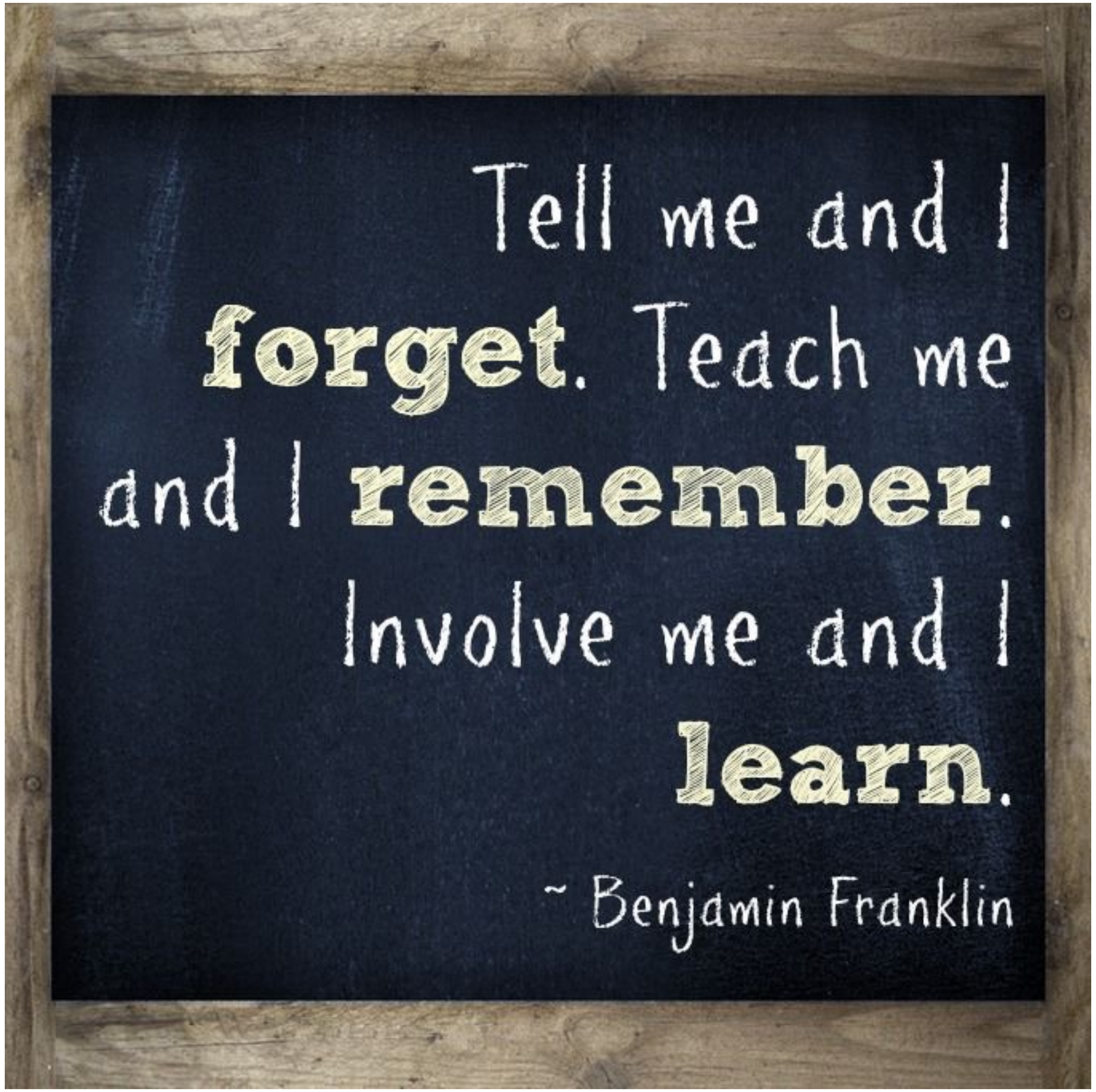
Packet delay

- what do “real” Internet delay & loss look like?
- **traceroute** program: provides delay measurement from source to router along end-end Internet path towards destination. For all *i:*
  - sends three packets that will reach router *i* on path towards destination (with time-to-live field value of *i*)
  - router *i* will return packets to sender
  - sender measures time interval between transmission and reply



* Do some traceroutes from exotic countries at www.traceroute.org

Thank You!


Tell me and I forget. Teach me and I remember. Involve me and I learn.
~ Benjamin Franklin

Network Address

Mac Address

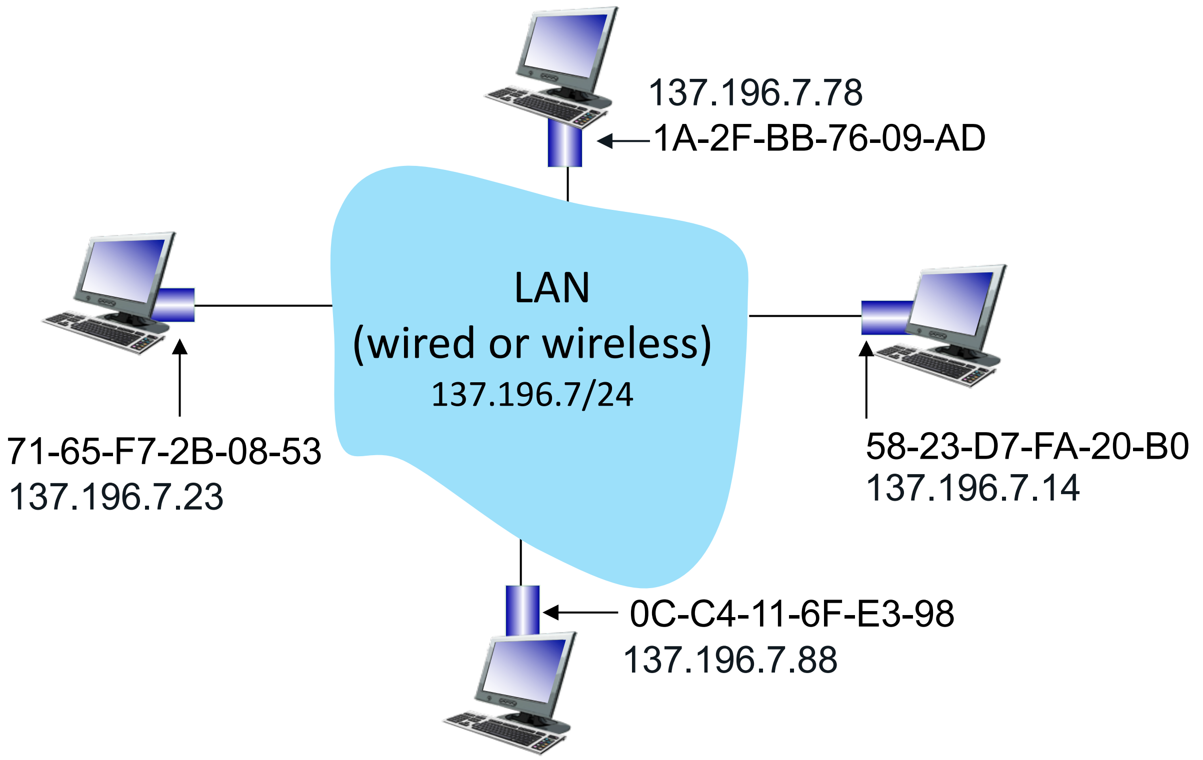
- 32-bit IP address:
  - *network-layer* address for interface
  - used for layer 3 (network layer) forwarding
  - e.g.: 128.119.40.136
- MAC (or LAN or physical or Ethernet) address:
  - function: used "locally" to get frame from one interface to another physically-connected interface
  - 48-bit MAC address 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)

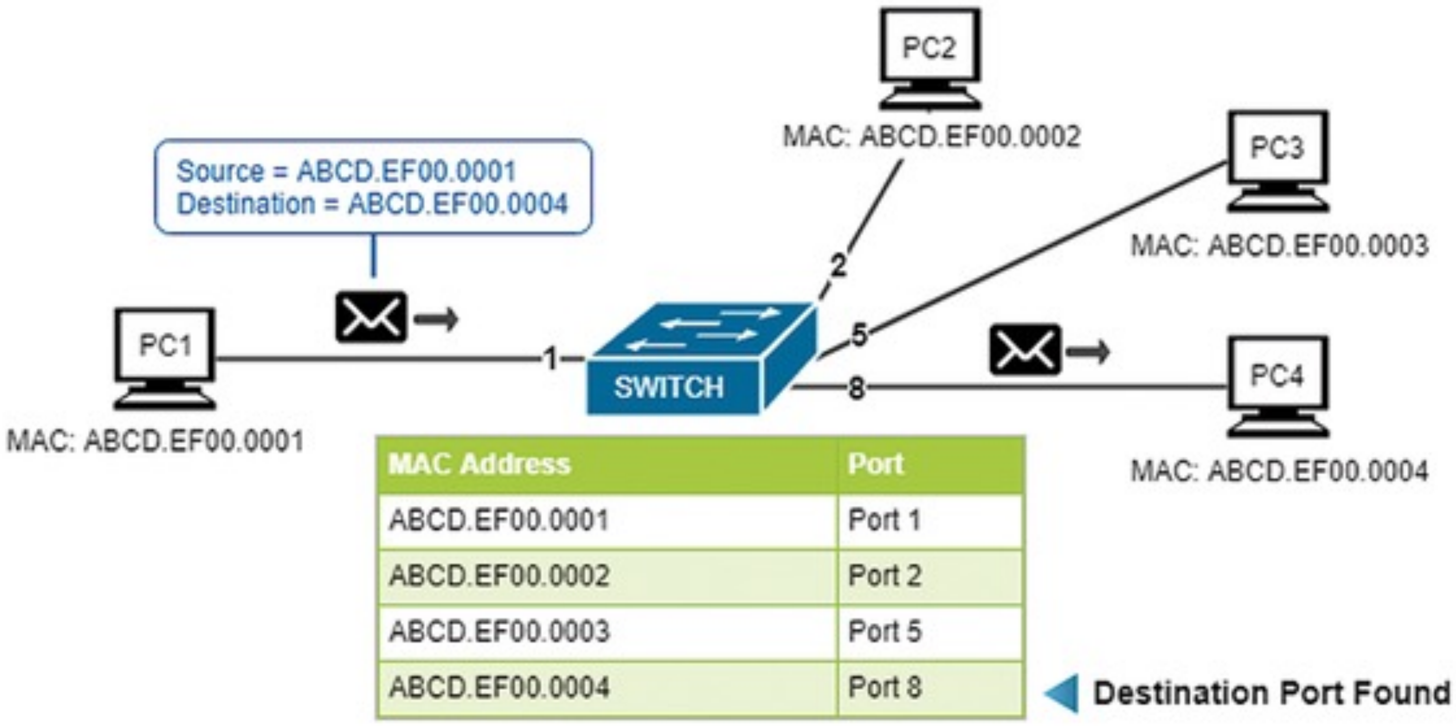
Mac Address

Each interface on LAN

- has unique 48-bit MAC address
- MAC address allocation administered by IEEE
- manufacturer buys portion of MAC address space



Mac Address Table



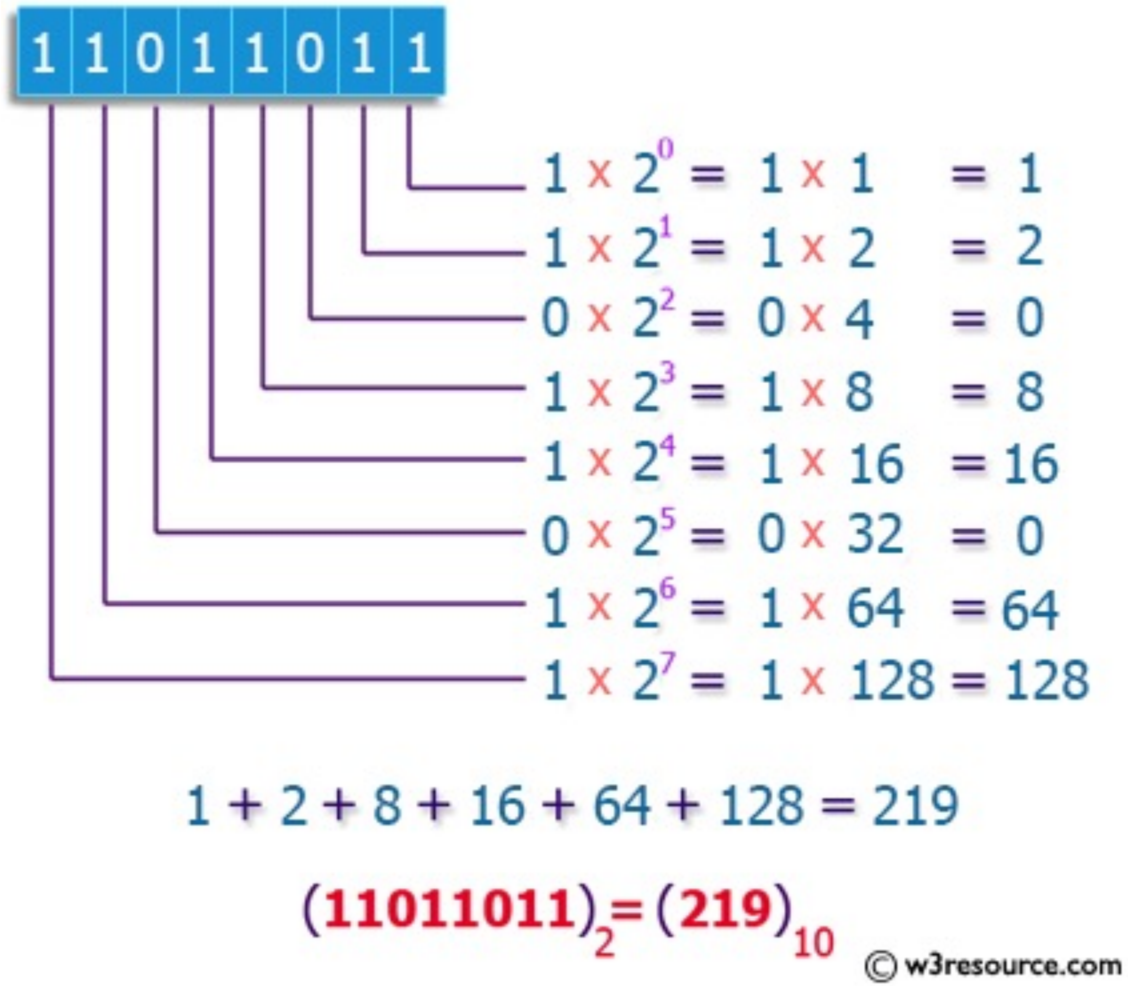

```
2960-1#show mac address-table
          Mac Address Table

-------------------------------------------

Vlan    Mac Address       Type        Ports
----    -----------       --------    -----
   1    001d.70ab.5d60    DYNAMIC     Fa0/2
   1    001e.f724.a160    DYNAMIC     Fa0/3
Total Mac Addresses for this criterion: 2
2960-1#
```

A MAC address table, sometimes called a Content Addressable Memory (CAM) table, is used on Ethernet switches to determine where to forward traffic on a LAN.

Switching Operation

1. The switch receives a frame.
2. The switch reads the source and destination MAC addresses.
3. The switch looks up the destination MAC address in its switching table. (Q: What if it's not found in the table?)
4. The switch forwards the frame to the port where the computer owning the MAC address is found.
5. The switching table is updated with the source MAC address and port information.

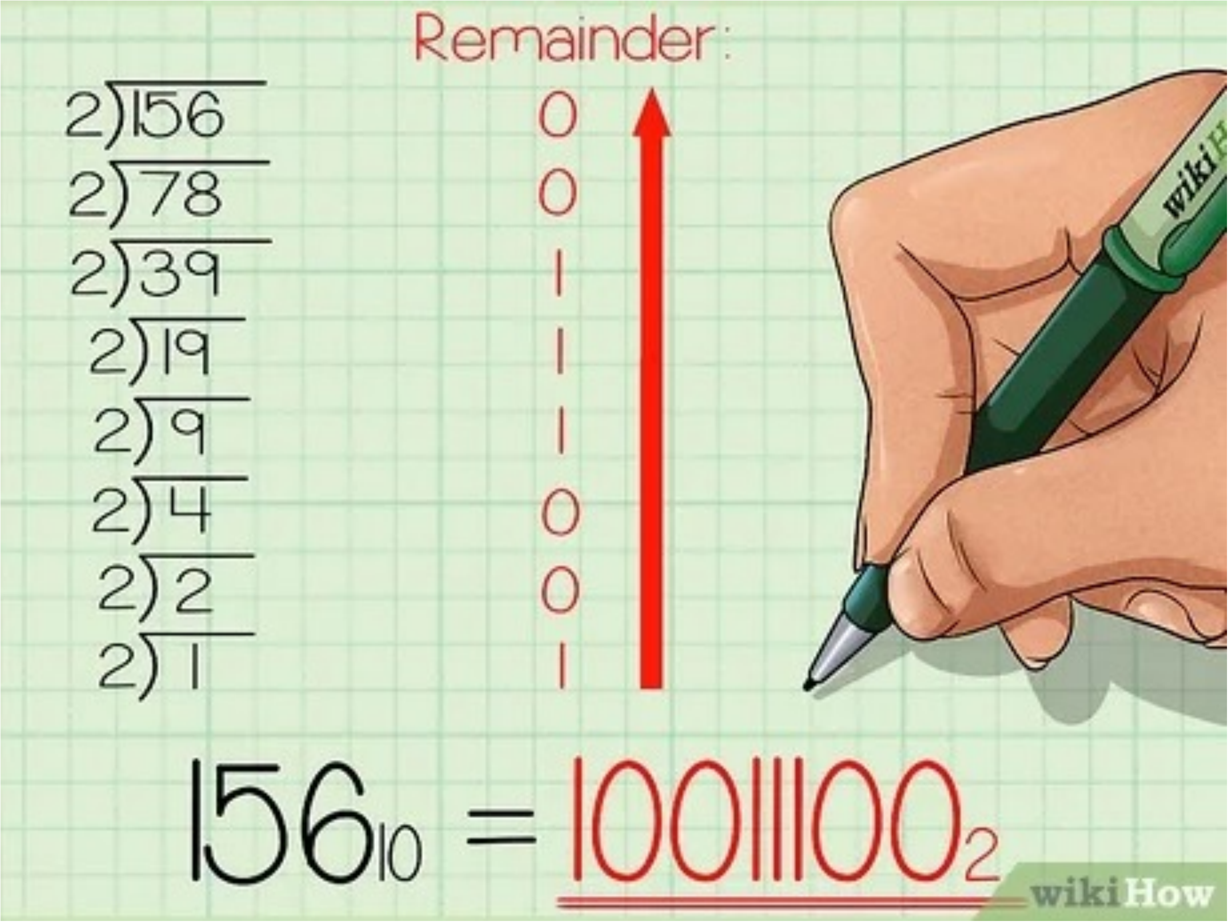
Binary To Decimal Conversion



128	64	32	16	8	4	2	1
1	0	0	1	0	0	1	0
0	1	1	1	0	1	1	1
1	1	1	1	1	1	1	1
1	1	0	0	0	1	0	1

Answers	Scratch Area	
146	128	64
	16	32
119	2	16
	146	4
		2
		1
		119

Decimal to Binary Conversion



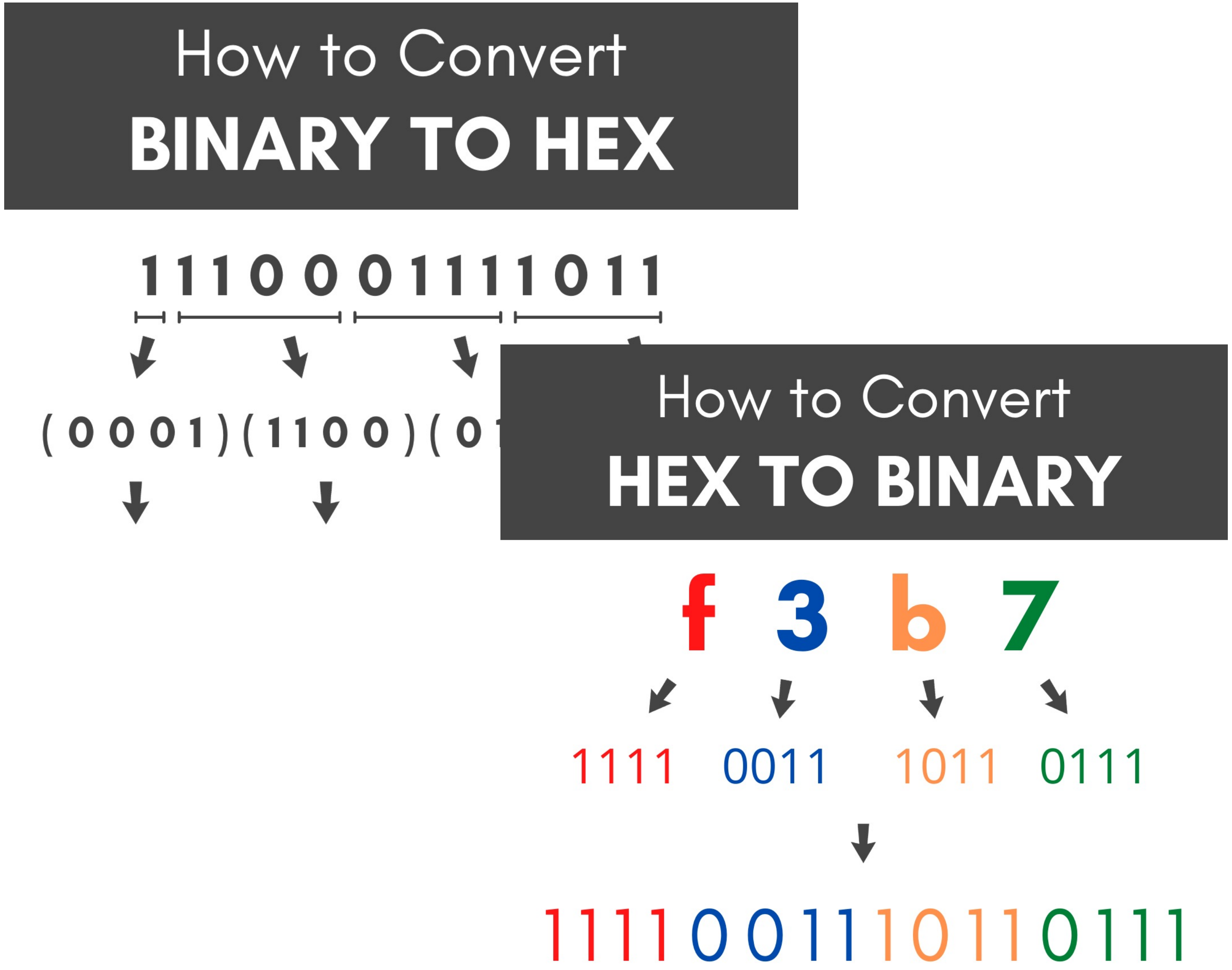
128	64	32	16	8	4	2	1	= 255
1	1	1	0	1	1	1	0	238
0	0	1	0	0	0	1	0	34

Scratch Area

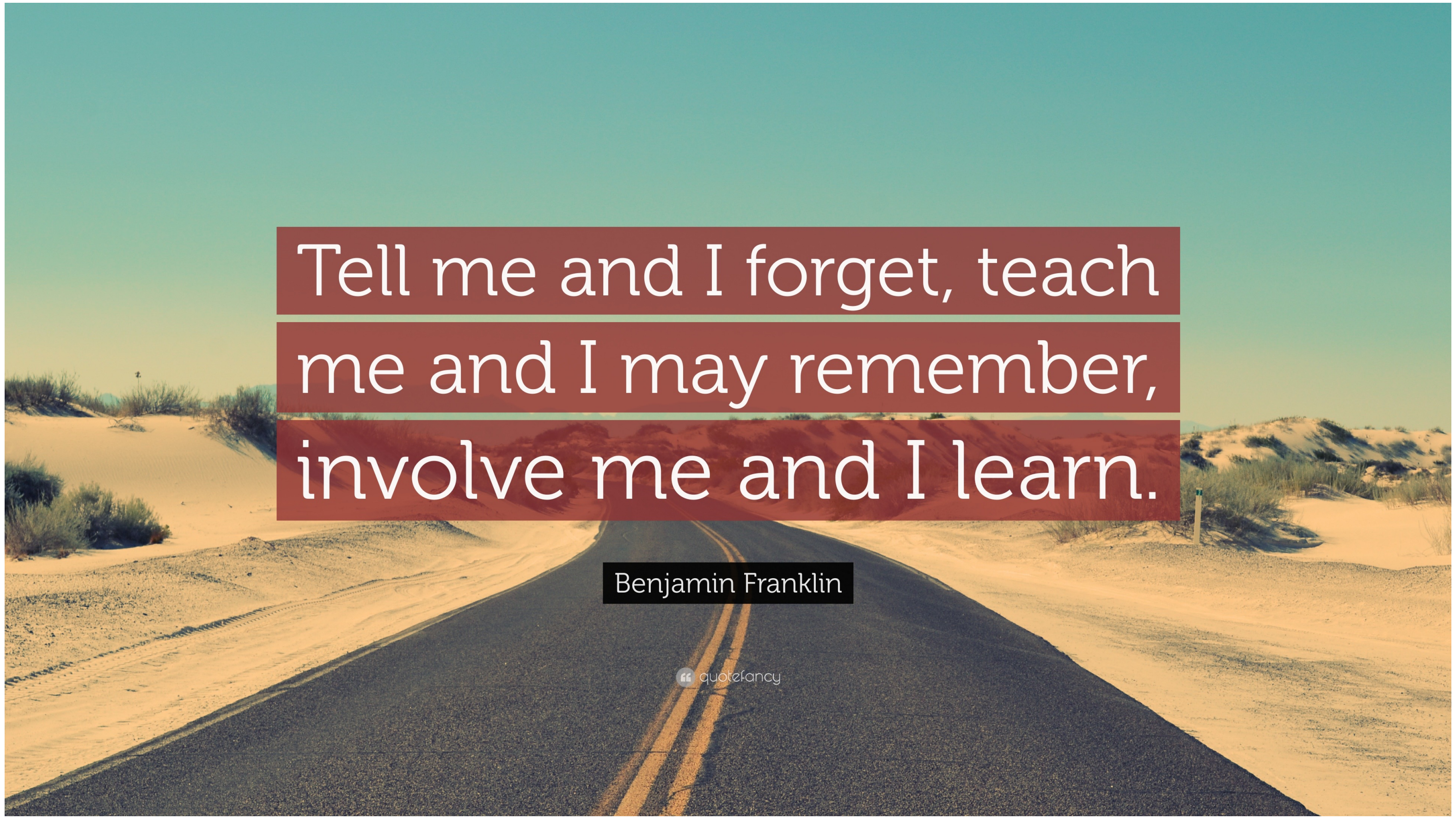
238	34
−128	−32
110	2
−64	−2
46	0
−32	
14	
−8	
6	
−4	
2	
−2	
0	

Decimal, Binary, Hexadecimal

Decimal (Base 10)	Binary (Base 2)	Hexadecimal (Base 16)
0	0000	0
1	0001	1
2	0010	2
3	0011	3
4	0100	4
5	0101	5
6	0110	6
7	0111	7
8	1000	8
9	1001	9
10	1010	A
11	1011	B
12	1100	C
13	1101	D
14	1110	E
15	1111	F



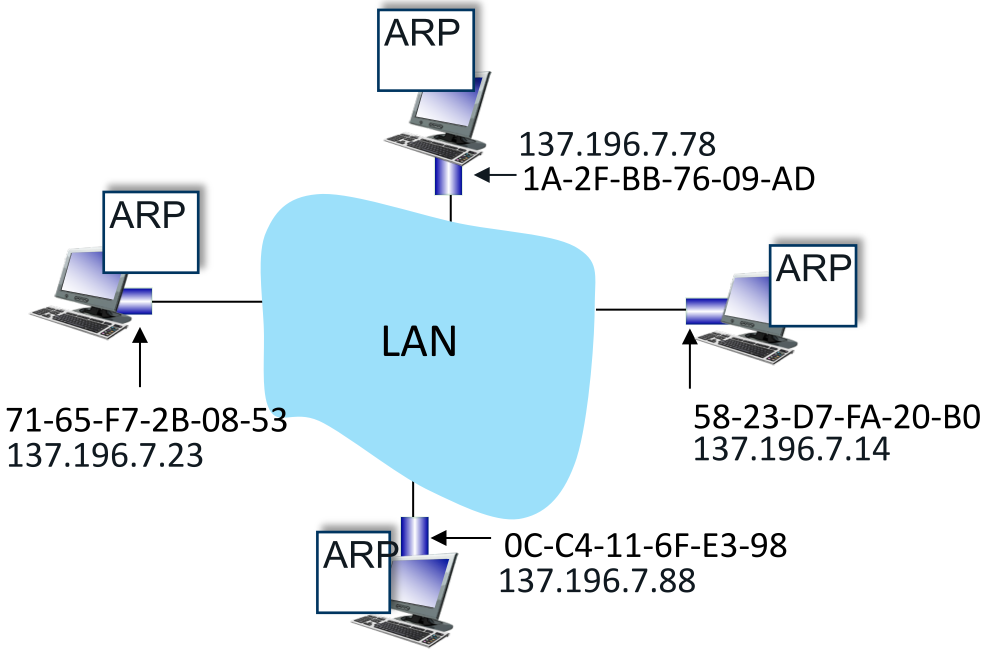
Thank You!


Tell me and I forget, teach
me and I may remember,
involve me and I learn.
Benjamin Franklin
quotefancy

Address Resolution Protocol (ARP)

Address Resolution Protocol (ARP)

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)

Address Resolution Protocol (ARP)

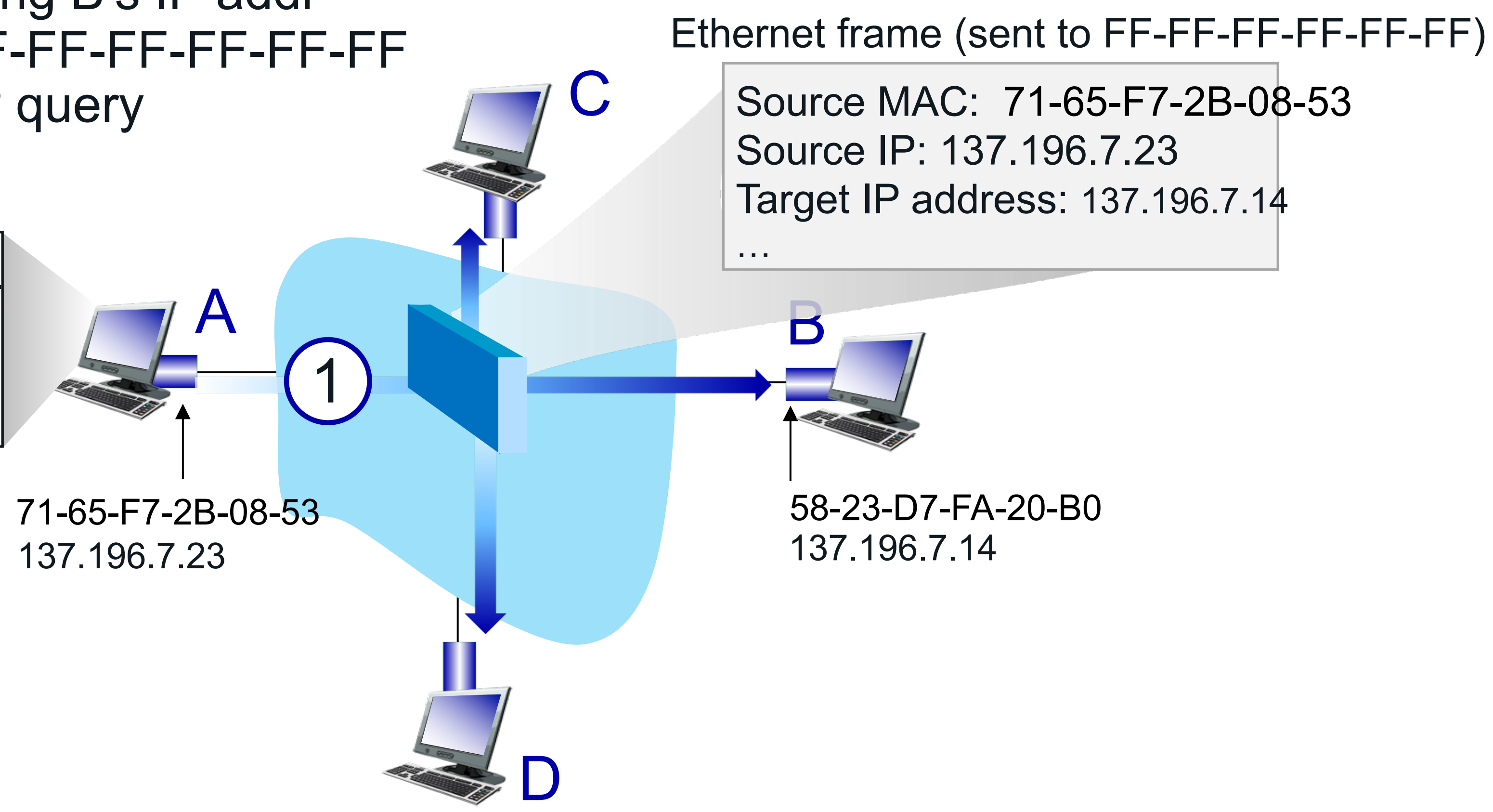
example: A wants to send datagram to B

- 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

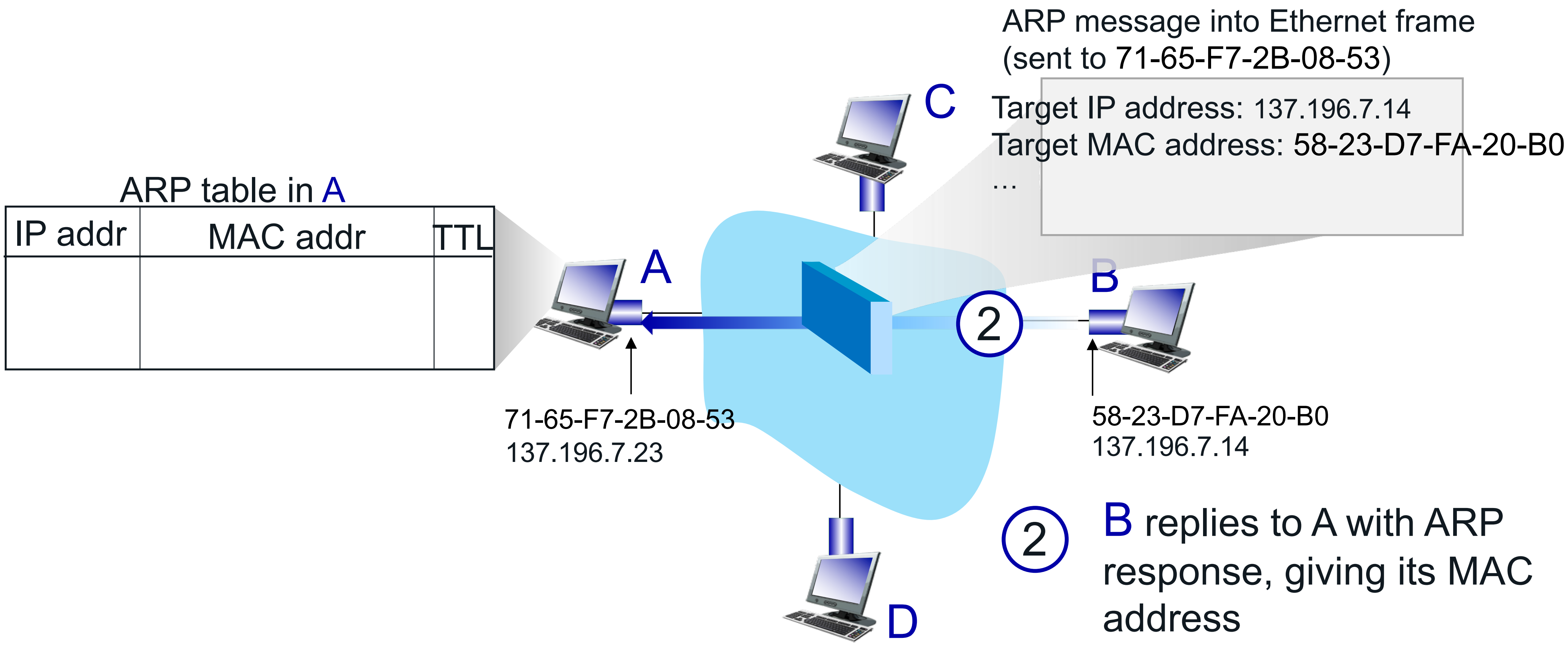
IP addr	MAC addr	TTL



Address Resolution Protocol (ARP)

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



2 B replies to A with ARP response, giving its MAC address

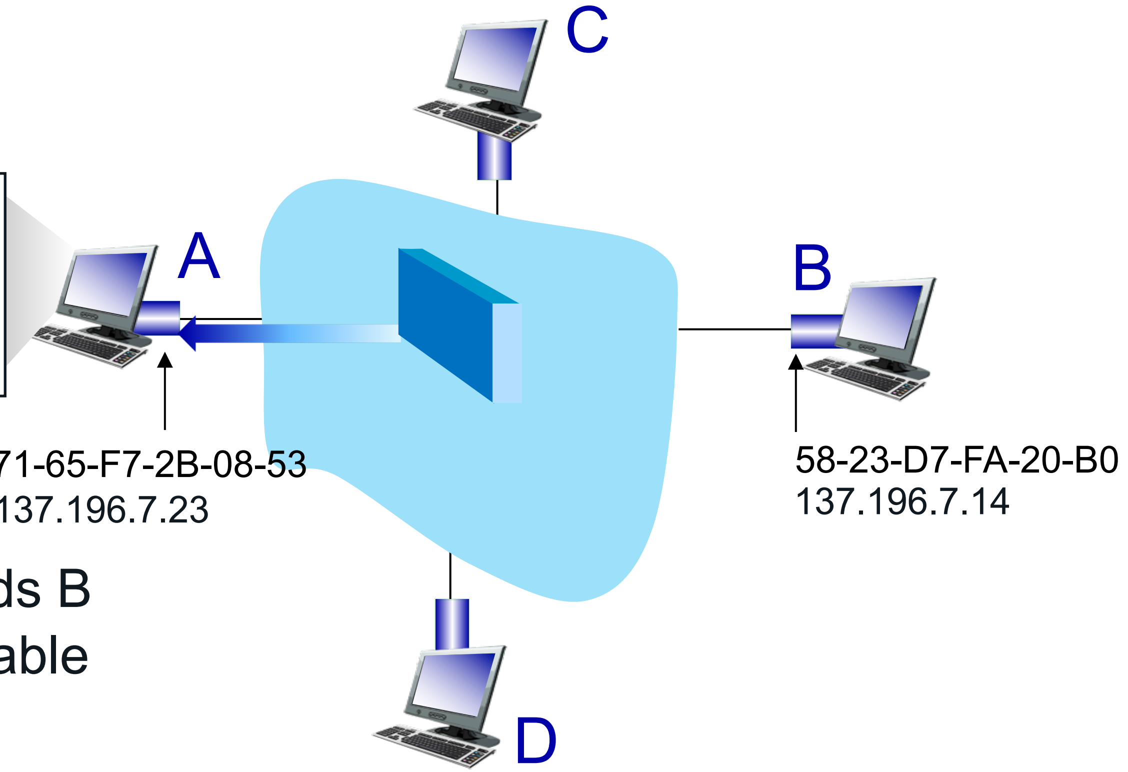
Address Resolution Protocol (ARP)

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

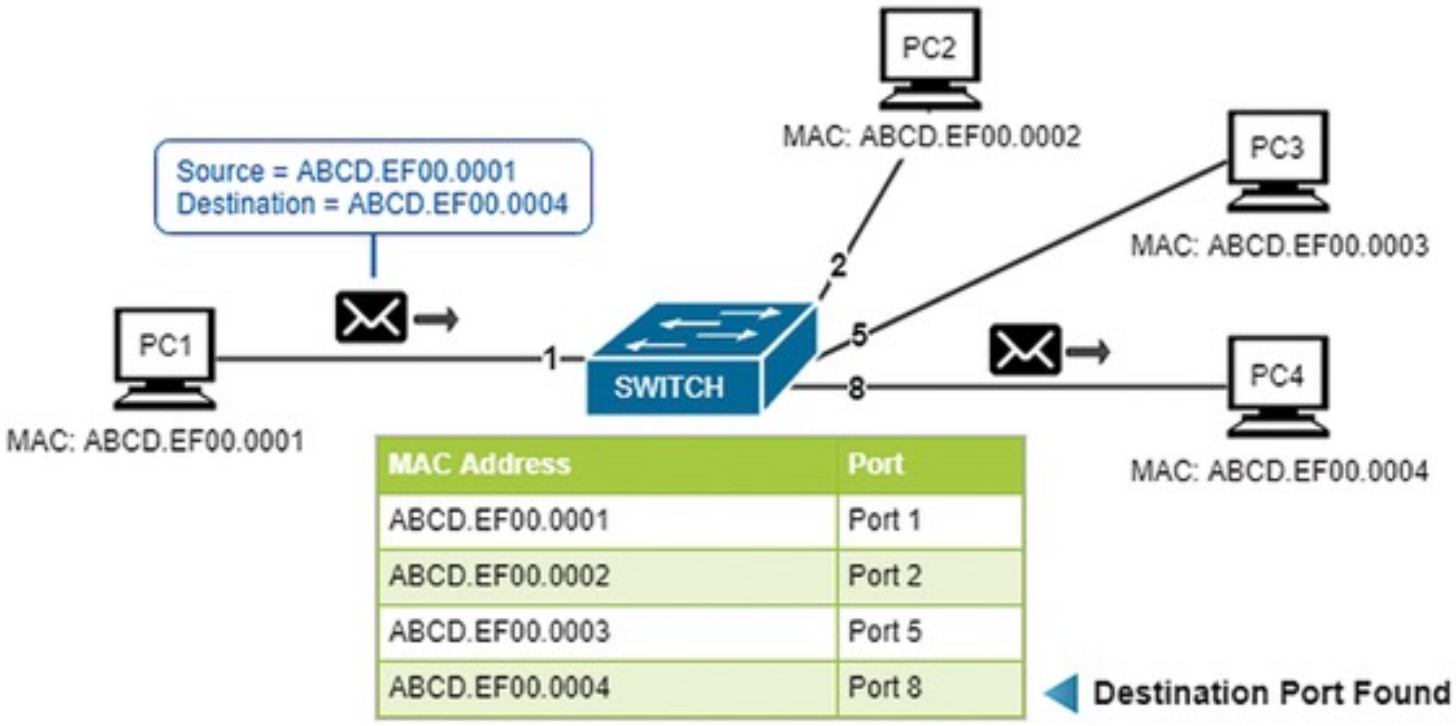
ARP table in A

IP addr	MAC addr	TTL
137.196. 7.14	58-23-D7-FA-20-B0	500



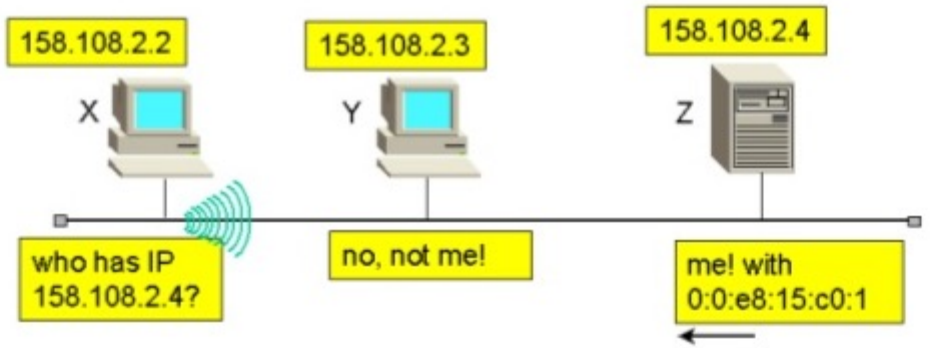
3 A receives B's reply, adds B entry into its local ARP table

Switches – ARP tables (link between MAC and IP)



MAC Address	Port
ABCD.EF00.0001	Port 1
ABCD.EF00.0002	Port 2
ABCD.EF00.0003	Port 5
ABCD.EF00.0004	Port 8

```
C:\Windows\system32\arp -a
Interface: 192.168.1.100 --- 0x11
  Internet Address      Physical Address      Type
  192.168.1.200         00-0c-29-92-81-3e     dynamic
C:\
```

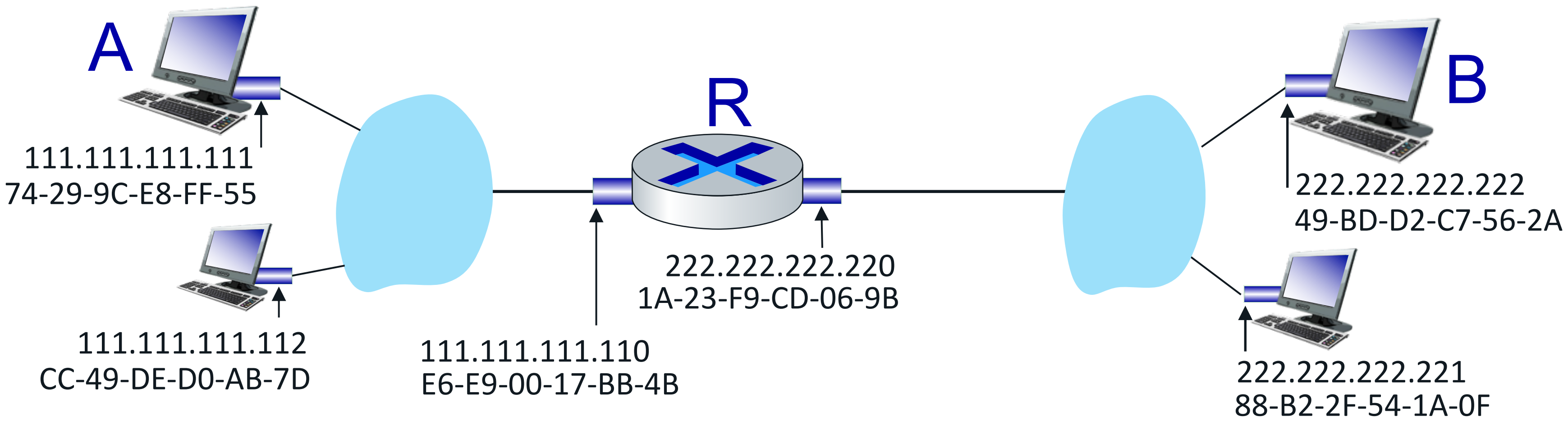


The Address Resolution Protocol (ARP) is designed for resolving addresses. To tie together the data link (Layer 2) layer and the network (Layer 3) layer, a mechanism must exist that maps data-link layer addressing to network layer addressing; this mechanism is ARP.

Routing to Another Subnet

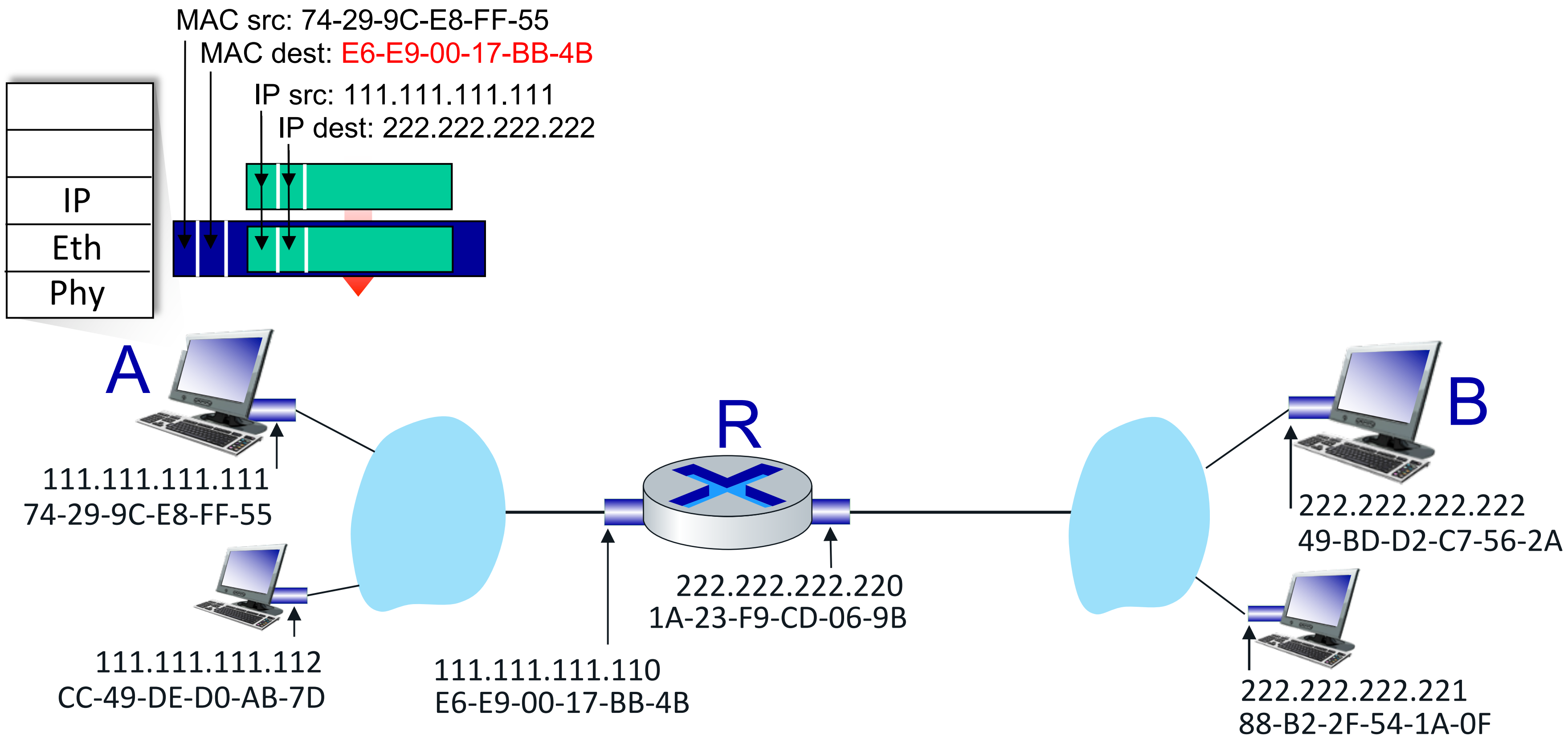
walkthrough: sending a datagram from *A* to *B* via *R*

- 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?)



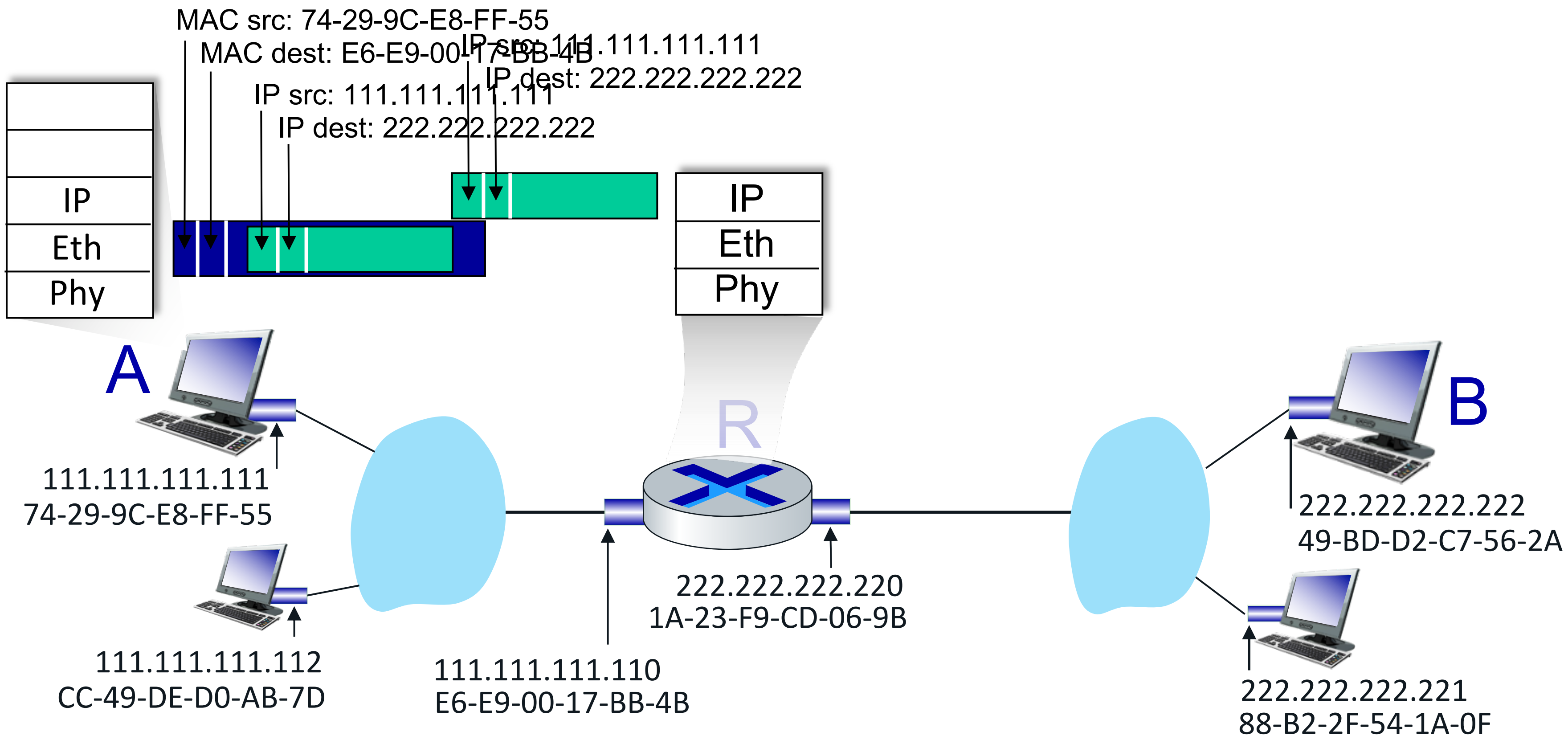
Routing to Another Subnet

- 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



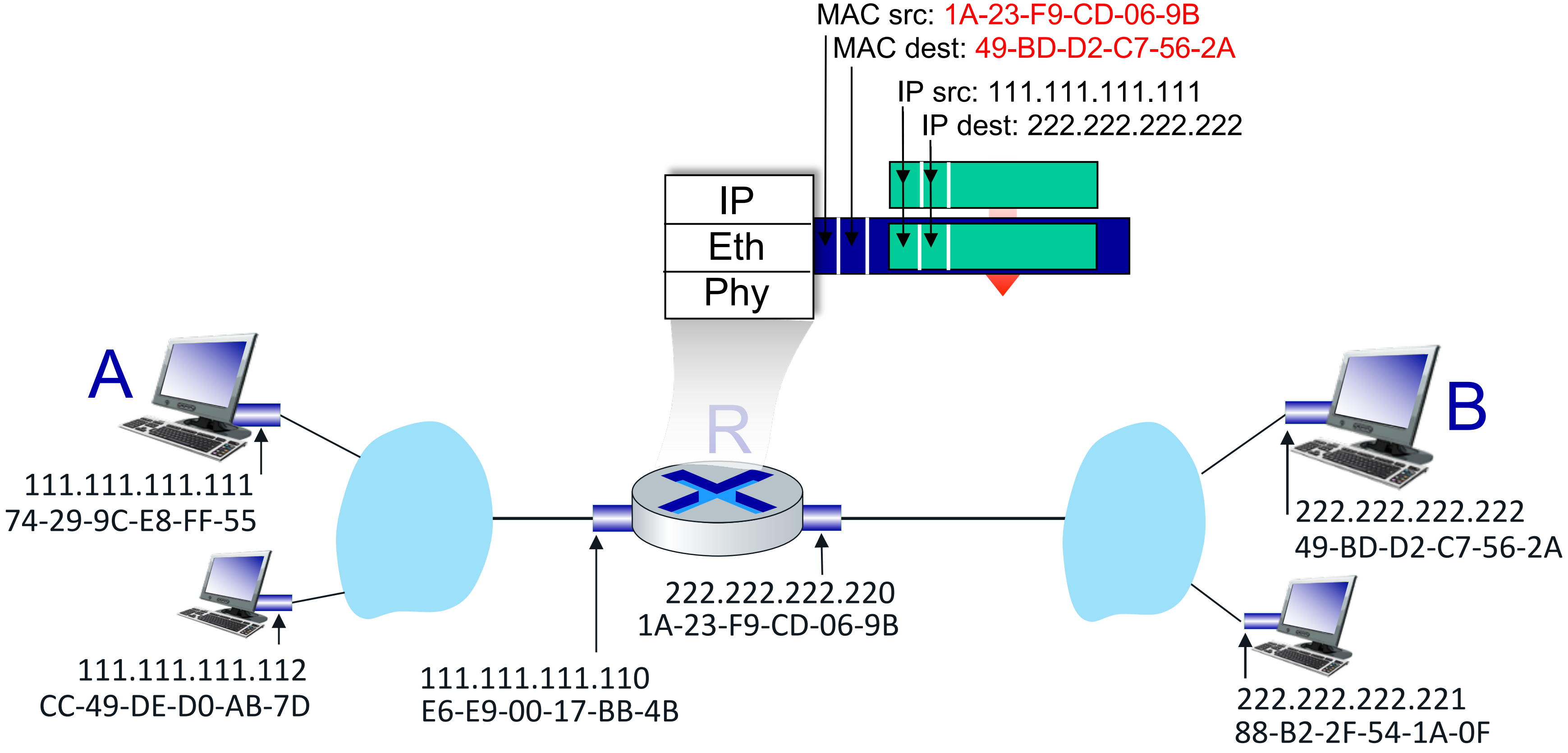
Routing to Another Subnet

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



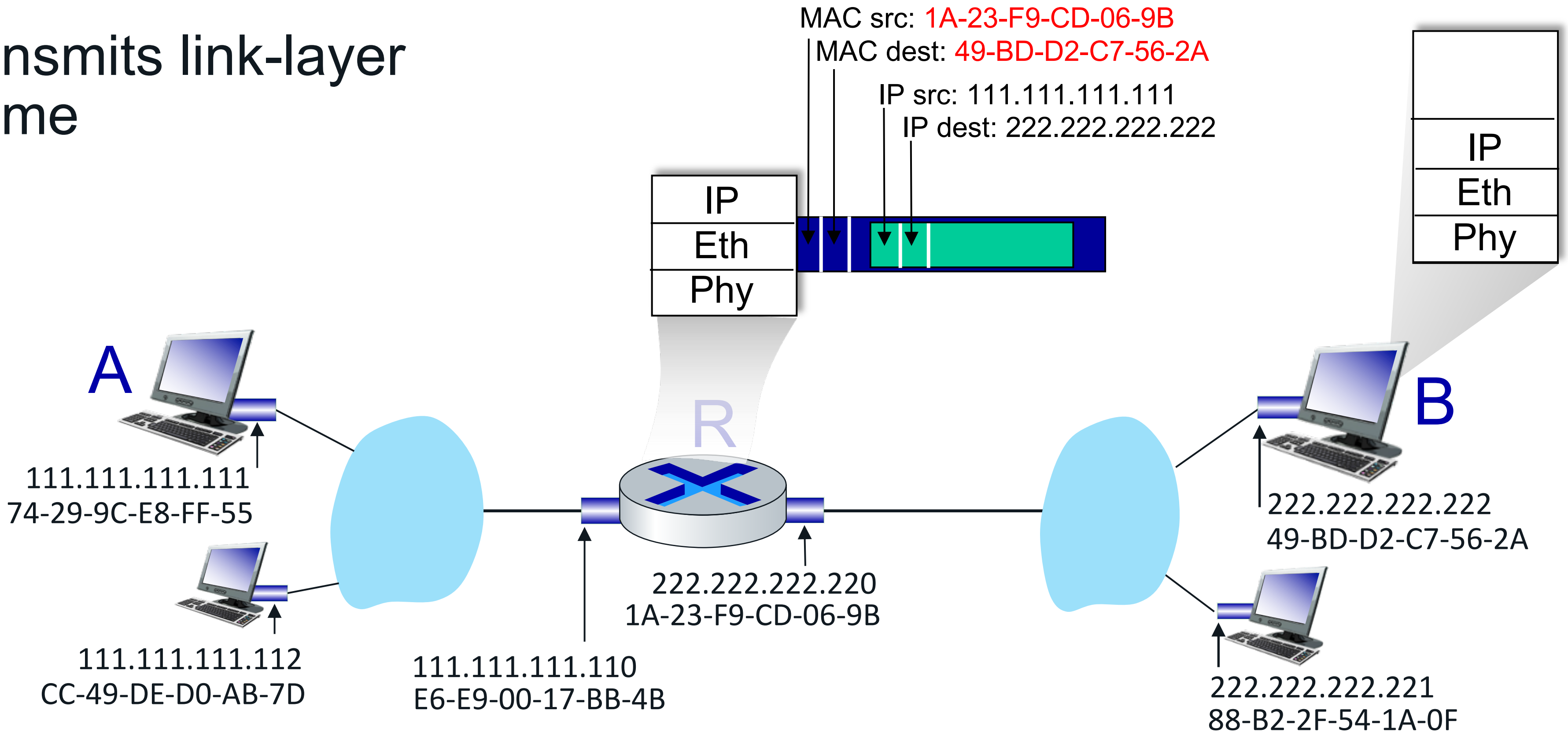
Routing to Another Subnet

- 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



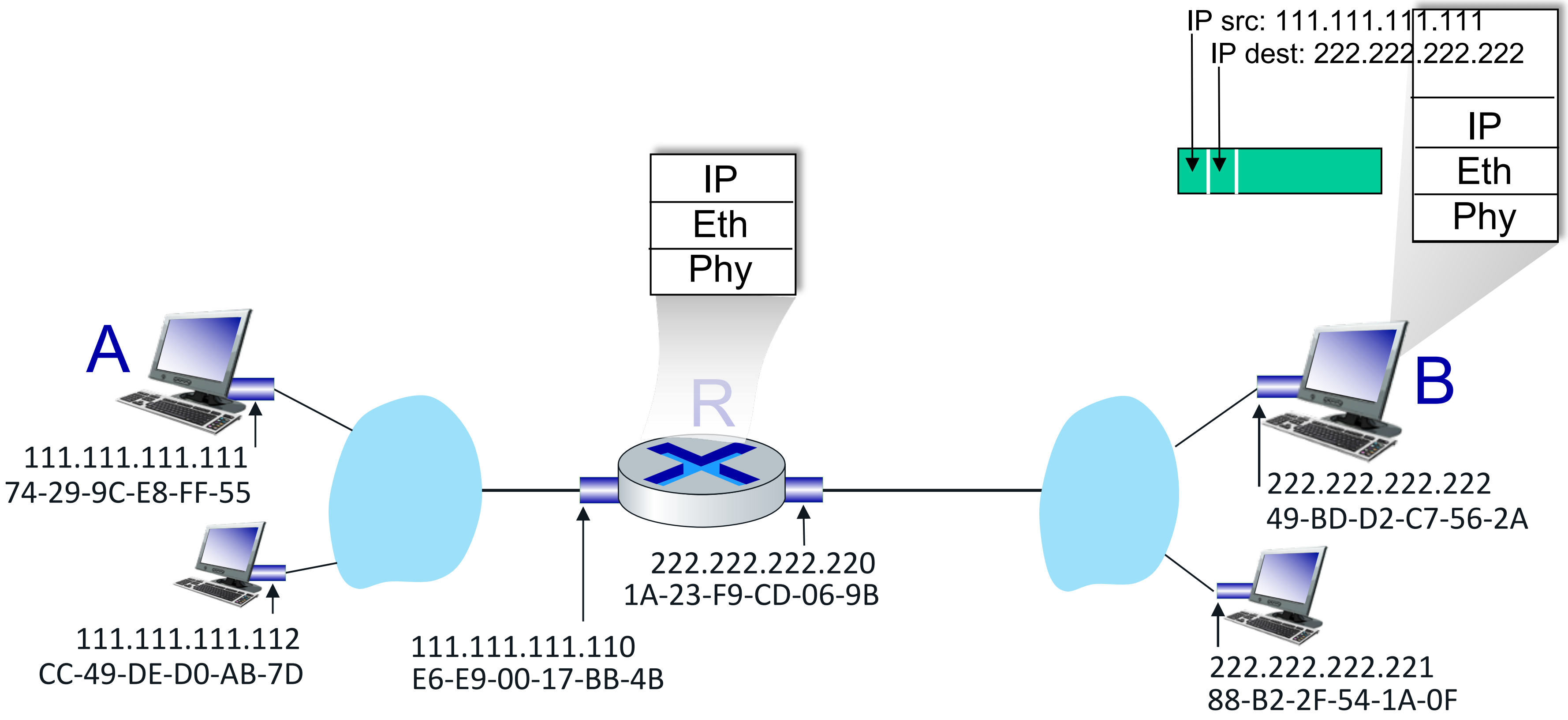
Routing to Another Subnet

- 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

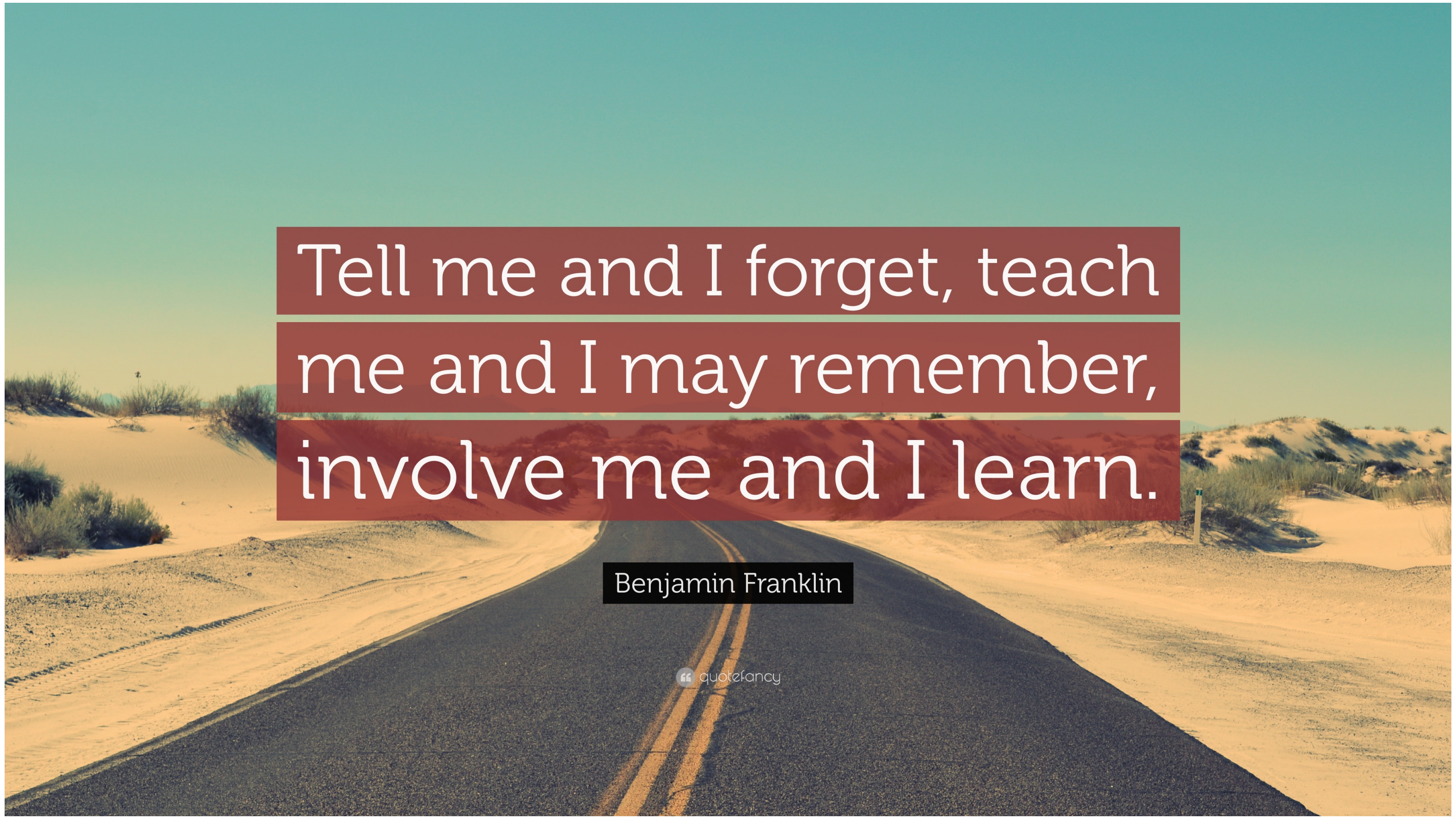


Routing to Another Subnet

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



Thank You!


Tell me and I forget, teach me and I may remember, involve me and I learn.
Benjamin Franklin
quotefancy